National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

NL-339 (r. 88/04) c

Canada

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-55518-1

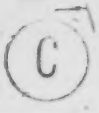
Canada

THE UNIVERSITY OF ALBERTA

A RISK ANALYSIS OF LOADS INDUCED BY WET SNOW ACCRETION ON TRANSMISSION LINES

BY

MARK ALLAN BOURASSA



A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

METEOROLOGY

DEPARTMENT OF GEOGRAPHY

EDMONTON, ALBERTA

FALL 1989

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Mark Allan Bourassa

TITLE OF THESIS: A RISK ANALYSIS OF LOADS INDUCED BY WET SNOW ACCRETION ON TRANSMISSION LINES

DEGREE: MASTER OF SCIENCE

YEAR THIS DEGREE GRANTED: FALL 1989

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

M. Mark A. Bourassa

11658 - 72 Ave.
Edmonton, Alberta
T6H 5H1

Date: October 11, 1989

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled A RISK ANALYSIS OF LOADS INDUCED BY WET SNOW ACCRETION ON TRANSMISSION LINES

submitted by MARK ALLAN BOURASSA

in partial fulfilment of the requirements for the degree of MASTER OF SCIENCE.

.....
(Supervisor) Robert B. Charlton

.....

.....
D. Wiens

.....
J. D. Wilson

Date: Oct. 11, 1989

Abstract

A risk analysis was performed on transmission line loads due to annual extreme wet snow accretions. Both horizontal and vertical loads were simulated using a existing model of wet snow accretion. The model requires values of five meteorological variables throughout the precipitation event: air temperature, relative humidity, precipitation rate, wind speed, and wind direction. Calculations, based on twenty years of data from CFB Namao, in central Alberta, Canada, gave the annual frequency of wet snow events, the duration of these events, and the five meteorological variables needed for the accretion model. The annual maximum loads were used in extreme value analyses to determine the relationships between the extreme loads and their mean return times. Finally risk analyses were used to determine the structural strengths of transmission lines and towers needed for these structures to have a specified chance of lasting a specified lifetime.

Table of Contents

Chapter		Page
I.	INTRODUCTION	1
1.1	Climatology of wet snow	1
1.2	Transmission line engineering	3
1.3	Previous and current research	4
1.3.1	Laboratory research	4
1.3.2	Mathematical models	8
1.3.3	Prevention of wet snow accretions	13
II.	THEORY OF WET SNOW ACCRETION	16
2.1	The formation of wet snow in the atmosphere	17
2.2	The accretion of wet snow on transmission lines	31
2.3	Modelling theory	36
III.	THEORY OF RISK ANALYSIS	40
3.1	Extreme value analysis	45
IV.	RESULTS AND DISCUSSION OF SURFACE AND RISK ANALYSES	49
4.1	Results of the analysis of surface input parameters	50
4.1.1	Visibility as a measure of precipitation rate	64
4.1.2	Estimation of gust speed	67
4.1.3	Initial values of the meteorological parameters	71
4.1.4	Persistence	75
4.1.5	Time-dependent trends	81
4.1.6	Statistics of hourly changes	84
4.2	Results of a risk analysis on extreme line loads	88
4.3	Current design standards	99
V.	SUMMARY AND RECOMMENDATIONS	101
5.1	Recommendations	107
VI.	Bibliography	109
VII.	APPENDICES	111
7.1	Appendix A - Proofs for equations in risk analysis	111
7.2	Appendix B - Melting layers	113
7.3	Appendix C - Temporal trends in the mean hourly changes	117
7.4	Appendix D - Listing of annual extremes	119
7.5	Appendix E - Computer programs	122
7.5.1	Meteorological variable analysis programs	122
7.5.1.1	Function UAPT	137
7.5.1.2	Function LAYER	138
7.5.1.3	Function UATEST	139
7.5.1.4	Subroutine DSTRBN	141
7.5.1.5	Subroutine SETVAL	143

7.5.1.6 Subroutine IDENT 147
7.5.1.7 Subroutine SPEAK 150
7.5.1.8 Subroutine VARABS 160
7.5.1.8 Subroutine PREAVE 165
7.5.1.9 Subroutine AVERAG 165
7.5.1.10 Subroutine HISTO 171
7.5.1.11 Subroutine MINIMA 175
7.5.1.11 Subroutine STATS 176
7.5.1.12 Program MELT 180
7.5.1.13 Function EXTRAP 184
7.5.2 Extreme annual accretion model programs 184
7.5.2.1 Program EXTREMES 184
7.5.2.2 Subroutine ACCRETE 187
7.5.2.3 Function SPCBET 193
7.5.2.4 Subroutine OMNICYL 194
7.5.2.5 Subroutine ROTATE 202
7.5.2.6 Subroutine SMOOTH 204
7.5.2.7 Subroutine XTINPT 206
7.5.3 Statistical programs and subroutines 207
7.5.3.1 Subroutine CURFIT 207
7.5.3.2 Program EVA 214
7.5.3.3 Program GAUSS 217
7.5.3.4 Program KRON 220
7.5.3.5 Program SORT 221

List of Tables

Table	Description	Page
1	Relative Humidity Regimes	28
2	Risks	42
3	Minimum Designed Return Periods	44
4	Snow Intensity vs. Average Hourly Snowfall	66
5	The Relation of Snow Intensity to Visibility	66
6	Correlation Coefficients Between Initial Values of the Meteorological Parameters	72
7	Distribution Statistics of Initial Conditions	73
8	Mean Likelihood of Persistence	76
9	Time Dependence of Hourly Changes Equal To Zero	77
10	Hourly Changes in Wind Direction Equal to Zero	78
11	Fraction of Hourly Changes Equal to Zero	79
12	Time-Dependent Trends	82
13	Time-Dependent Trends	83
14	Correlations Coefficients of Hourly Changes	85
15	Distribution Statistics of Hourly Changes	86
16	Extreme Value Statistics	89
17	Constants 'a' and 'u'	90
18	Extreme Values as a Function of Return Period	92
19	Extreme Values as a Function of Return Period	93

20	Sufficient Design Mass Tolerance	96
21	Sufficient Design Vertical Load Tolerance	97
22	Sufficient Design Average Wind Tolerance	98
23	Sufficient Design Gust Tolerance	99
24	Load Tolerances of Transmission Lines	100

List of Figures

Figure	Description	Page
1	Cross Sections of Wet Snow Accretions	2
2	Techniques of Accretion Prevention on Transmission Lines	15
3	Surface Critical Relative Humidity	23
4	Relative Humidity Regimes	27
5	Forces Acting on the Accretion	33
6	Distribution of the Number of Annual Potential Wet Snow Events	51
7	Distribution of the Duration of Potential Wet Snow Events	53
8	Cumulative Probability Distribution of Duration	56
9	Frequency Distributions of the Initial Values of Air Temperature and Relative Humidity	57
10	Frequency Distribution of the Initial Values of Wind Direction and Wind Speed	58
11	Frequency Distributions of the Initial Values of Visibility	59
12	Frequency Distributions of the Hourly Changes Air Temperature and Relative Humidity	61
13	Frequency Distribution of the Hourly Changes in Wind Direction and Wind Speed	62
14	Frequency Distributions of the Hourly Changes in Visibility	63
15	Precipitation Rate as a Function of Visibility	65
16	Transmission Line Tower	70

Table of Symbols and Operators

a = one of the constants describing the relationship between extreme values and reduced variates

Antilog = antilog to the base ten

c = fraction of snow that must melt to maintain a heat balance [fraction]

C_p = specific heat capacity of air at a constant pressure [J K^{-1} kg^{-1}]

dir = wind direction [degrees]

dir_t = mean hourly change in wind direction, as a function of the number of hours since the onset of the precipitation event [degrees]

D = molecular diffusivity of water vapour in air [m^2 s^{-1}]

D_c = diameter of the cylinder [m]

e_a = vapour pressure in the ambient air [kPa]

e_s = saturation vapour pressure over ice [kPa]

exp = antilog to the base e

F_D = ventilation coefficient for diffusion [0]

F_K = ventilation coefficient for conduction [0]

I = electrical current [A]

k_v = von Karman's constant [0]

K = thermal conductivity [J m^{-1} s^{-1} K^{-1}]

ln = natural log (log to the base e)

L_f = specific latent heat of fusion [J kg^{-1}]

L_s = specific latent heat of sublimation [J kg^{-1}]

L_v = specific latent heat of vaporization [J kg^{-1}]

Log = log to the base ten

LWF = liquid water fraction (by mass) of an accretion or snowflake [fraction]

p = probability of exceedance [fraction]

P = pressure [kPa]

P_t = mean hourly change in pressure, as a function of the number of hours since the onset of the precipitation event [kPa]

P( ) = probability that the condition within the brackets is true [fraction]

Pr = Prandtl number [0]

PR = precipitation rate [mm water equivelent / hour]

Q_e = rate of heat exchange due to evaporation or condensation [J/s]

Q_m = rate of heat exchange due to melting of snow [J/s]

Q_J = rate of heat exchange due to Joule heating [J/s]

Q_v = rate of heat exchange due to convection to the ambient air (ventilation) [J/s]

r = risk: the probability of an event of magnitude greater than a specific value occurring within a specific time [fraction]

r_a = mixing ratio of the air [kg water vapur/kg air]

r_s = saturation mixing ratio [kg water vapur/kg air]

R = electrical resistance per unit length [ohms/m]

R_v = specific gas constant of water vapour [J kg^{-1} K^{-1}]

RH = relative humidity [percent]

RH_c = critical relative humidity [percent]

RH_{cs} = surface critical relative humidity [percent]

RH_t = mean hourly change in relative humidity, as a function of the number of hours since the onset of the precipitation event [%]

RSD = a randomly generated number of standard deviations

S = sticking efficiency [fraction]

Sc = Schmidt number [0]

t = time, usually in hours

t_a = temperature of ambient air [^oC]

T = return time

T_a = temperature of ambient air [K]

T_c = temperature of transmission line [K]

T_s = temperature of the surface of the snow flake [K]

T_{at} = mean hourly change in temperature, as a function of the number of hours since the onset of the precipitation event [K]

u = one of the constants describing the relationship between extreme values and reduced variates

u_* = friction velocity [m/s]

U = wind speed [m/s]

U_c = velocity component of the snowflake, perpendicular to the cylinder's axis [m/s]

U_t = mean hourly change in wind speed, as a function of the number of hours since the onset of the

precipitation event [m/s]

v_g = gust speed [m/s]

V = visibility [km]

V_t = mean hourly change in visibility, as a function of the number of hours since the onset of the precipitation event [km]

W = fall speed of snowflakes [m/s]

Y = reduced variate

z = height from the surface [m]

z_o = roughness length [m]

δ = Kronacker delta

Δ = change in the following variable

ρ_a = vapour density of ambient air [kg m^{-3}]

ρ_s = saturation vapour density of air with respect to ice [kg m^{-3}]

INTRODUCTION

The accretion of wet snow on transmission lines is a problem in many countries (Wakahama et al., 1977). The wet, sticky snow collects around a transmission line (see Fig 1). This increases its weight. It also increases its surface area, and hence the force applied by the wind. When accretion takes place over a large area, several kilometres of line can be brought down. For example, one storm in Saskatchewan brought down ninety-three towers (Wakahama et al., 1989). Damages can result in tens of millions of dollars in replacement costs. It can also cause power outages over large areas. In our society this can be life threatening. In an industrialized and electronically oriented society, prolonged power outages are more than a minor annoyance. Wet snow accretion is therefore a significant and costly problem.

1.1 Climatology of wet snow

Wet snow accretion is a problem wherever snow often falls at temperatures near zero degrees Celsius (Wakahama et al., 1977). Large bodies of water not only provide a source of moisture, but they also can moderate the temperature to values near zero degrees. Mountains provide a means for lifting moist air to cause precipitation (Finstad et al., 1988). Consequently the presence of mountains and large bodies of water increases the likelihood of wet snow events, but they are not necessary for it to occur.

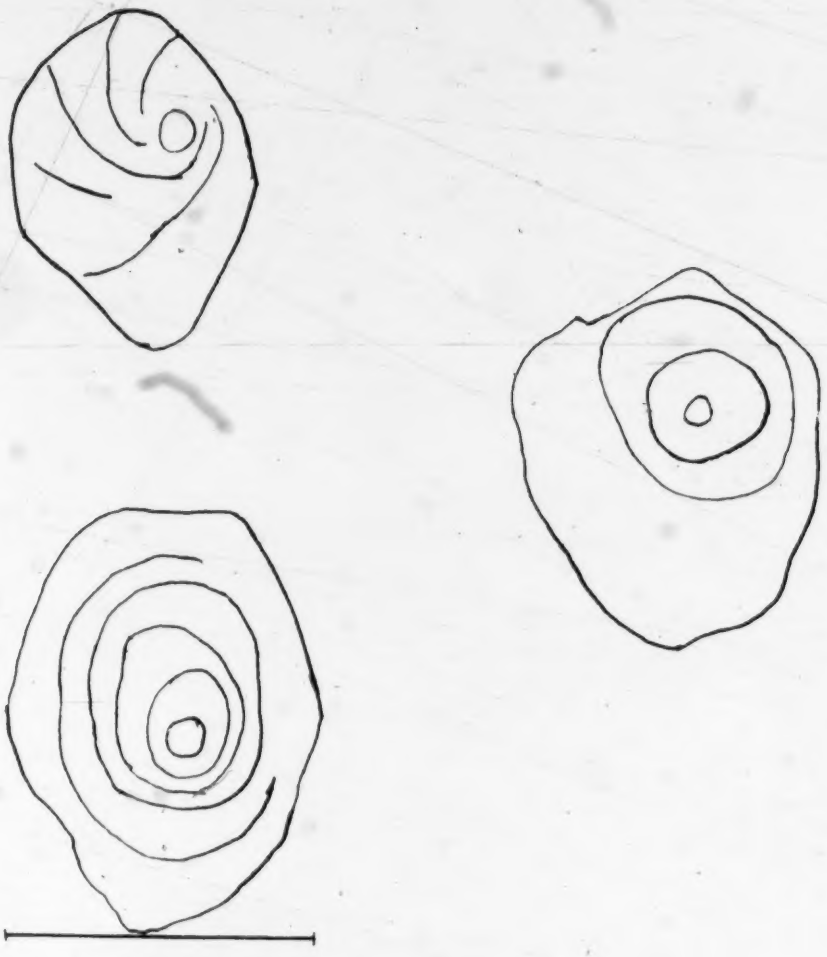


Figure 1 - Cross Sections of Wet Snow Accretions on Cylinders

The figures are sketches of the accretion patterns in actual wet snow accretions (adapted from Wakahama et al., 1977). Note that the accretions are not radially symmetrical.

Many nations have problems with wet snow. These include the United States, Canada, Norway, France, and Japan. The last four have encouraged some research into mitigation of the problem. Some locations as far inland as Alberta are likely to receive more than ten wet snow events each year (Lozowski et al., 1989). In Alberta wet snow events have a greater tendency to occur in early fall or late spring than at other times of the year (Lozowski et al., 1989). Approximately half the wet snow events occur in October and April. These are the months with the greatest number of hours with precipitation occurring simultaneously with a temperature near freezing.

1.2 Transmission line engineering

Transmission line conductors are typically several centimetres in diameter, and hundreds of metres in length between towers. This provides a large surface area on which wet snow can collect. The extra weight alone is seldom enough to down the lines. The wires are strong enough to hold hundreds of kilograms per metre of line. Poles and towers are secured in the ground so well that they will buckle or break before they are pulled from the ground. In a static situation, the forces on both sides of a pole or tower are, to a good approximation, symmetrical. Thus the net force is directed downwards along the axis of the pole or tower. Wind destroys this otherwise workable system by directing the net force away from the axis of the tower. With enough force perpendicular to the pole or tower, it

will buckle or break. Once one tower buckles, the lines exert unbalanced forces on the neighboring towers. This may create a wave of destruction down the line. A severe wet snow event can therefore destroy tens of towers and several kilometers of line.

1.3 Previous and current research

1.3.1 Laboratory research

Wet snow accretions have been studied in Japan since the 1930s. In 1953, Shoda published a detailed study of the growth process of wet snow accretions (Wakahama et al, 1977). He studied only the accretions associated with heavy, wet snowfalls and winds of less than 3 m/s. These conditions are typical of the coastal regions of central Honshu, facing the sea of Japan. Shoda found that wind gusts in excess of 3 m/s would dislodge the accretions from the transmission lines. The accretions studied by Shoda are a special case. They are not representive of the bulk of wet snow accretions throughout the world; most accretions occur with wind speeds on the order of 10 m/s.

It is very difficult to simulate wet snow in the laboratory. Man-made wet snow is a poor substitute for the flakes and water content associated with wet snow (Wakahama et al., 1977; Admirat et al, 1985b). The method Wakahama used for making artificial wet snow was to take snow (preferably fresh) from the ground, and to drop it through a three layered vibrating sieve to disaggregate the snowflakes, before they fell into a temperature controlled

wind tunnel. The snowflakes were made wet by being sprayed, before they hit the target wires, with water at a temperature of zero degrees Celsius. The air temperature was kept between $+1^{o}C$ and $+2^{o}C$. Time lapse photographs were used to examine the growth process of the accretion, and to study the trajectories of snowflakes approaching the accretion.

Wakahama, Kuroiwa, and Goto found that the accretion would rotate around the cylinder, and that the accretion was rarely blown away after it encircled the wire. They found that large accretions could occur at any wind speed. Usually the accretion would slide down the cylinder, taking the shortest route to the position where the horizontal components of the forces acting on the accretion were balanced about its center of mass. However, at high wind speeds aerodynamic lift might rotate the accretion upward over the top of the cylinder. When stranded cables (non-smooth cylinders) were used, they found that unless the wire rotated easily, the accretions would tend to be blown off the cylinder. The adhesive stress of the wet snow on the cylinder was examined and found to be as expected for an object the size of a snowflake completely coated by water: 19 ± 1 kPa. The adhesive stress was found to have such a value only when the spaces between snowflakes were filled with water. This was the case when the accretion had a liquid water faction (by mass) of 20% or greater.

Time lapse photography was used to examine the collision and sticking efficiencies of wet snowflakes

striking a cylinder. The collision efficiency is the ratio of the number of snowflakes that hit the cylinder, to the number of snowflakes that would hit the cylinder if it did not alter the air flow. The sticking efficiency is the ratio of the number of snowflakes that stick to the cylinder, to the number of snowflakes that hit the cylinder. Time lapse pictures showed that the trajectories of the snowflakes,before they hit the cylinder, were straight to a very good approximation. Consequently the collision efficiency for wet snowflakes is unity. Photographs also showed that, for a wire 4 cm in diameter, over 80% of the snowflakes rebounded when they hit the wire. Therefore the sticking efficiency was less than 20%. This fact is important, because the mass of the accretion is proportional to the sticking efficiency.

More recent research on the sticking efficiency of wet snowflakes has been performed by Admirat, Lapeyre, and Maccagnan (1985b). They sprayed grated snow into a temperature controlled wind tunnel. The snow was dry when it entered the wind tunnel. Heating by the ambient air caused some melting before the snow hit the target wire. No measurements of the liquid water fraction of the snowflakes were made. The relative humidity was kept between 85% and 95%. The wind speed, the precipitation rate, and the air temperature were easily varied. Experiments were made over a wide range of these variables. Admirat, Lapeyre, and Maccagnan's experimental results were compared to the

predictions of a numerical model developed by the same research group.

Admirat et al. observed that wet snow accretions had the same growth and rotational properties that were observed by Wakahama et al. (1977). Admirat et al. noted that, after the accretion had rotated 180° from its original position, its shape was approximately circular, with the cylinder off-center. They also found that the rate of rotation was almost constant over the first hour of the accretions. They examined the temperature of the conductor to confirm their assumption that the temperature of the accretion was 0°C. They found that regardless of the air temperature, the temperature of the line rapidly approached zero (in ten to fifteen minutes). They also examined the sticking efficiency of the snowflakes, and the density of the accretion, as functions of the air temperature and the precipitation rate. They fitted polynomials to their findings so that these parameters could be estimated within their accretion model. The polynomials are:

$$S = 1.759\ PR^{-1.509}\ T_a^4 - 10.638\ PR^{-1.282}\ T_a^3 + 34.898\ PR^{-1.277}\ T_a^2 - 11.532\ \exp(-0.612\ PR)\ T_a - 0.485\ \ln(PR) + 1.607,$$

$$\rho = -108.2 + 11.5\ PR + 164.7\ T_a - 5.4\ PR\ T_a,$$

where

PR = precipitation rate [mm water equivalent / hour]

S = sticking efficiency [fraction]

T_a = temperature of ambient air [K]

ρ = density of the wet snow

Predictions of the mass of accretions by a model using these polynomials compared favorably with the observed characteristics of accretions in additional wind tunnel experiments.

Admirat et al. also examined the effects of Joule heating on the growth of wet snow accretions. Joule heating is the production of heat by an electric current passing through a resistive conductor. The power output of Joule heating is equal to the electrical resistance times the square of the current. This is of interest because transmission lines carry a current and consequently produce heat through Joule heating. Admirat et al. found that line heating had no effect on the rotation of the accretion, and no effect on the size of the accretion for precipitation rates greater than 15 (water equivalent) mm/hr. At lower precipitation rates the accretions would be too wet, and consequently they fragment and fall off the line. A precipitation rate of 15 (water equivalent) mm/hr is a heavy precipitation rate. This implies that wet snow accretion is unlikely to occur on a hot transmission line. The effect of Joule heating is to raise the liquid water content of the accretions.

1.3.2 Mathematical models

Mathematical models are useful research tools because they allow experiments to be performed on paper or by a

computer. Ice accretion models had been devised as early as the 1950s (Lozowski and Gates, 1987). Ice accretion is in some respects similar to wet snow accretion. Both model the accretion of airborne particles on transmission lines. However relatively few models have been constructed for wet snow accretion. Wet snow accretion models to date have been based on either the thermodynamics or the mechanics of the accretion process, but not on both.

The model produced by Admirat, Lapeyre, and Maccagnan (1985a) is a thermodynamical model. It ignores the mechanics of the problem, and treats the accretion as radially symmetrical at all times. While this is not the case in reality, it is certainly workable as a first approximation. The model separates the accretion process into three stages: formation, growth, and collapse. The formation stage is a pre-accretion stage. It determines, relative to the onset of the precipitation, when the growth process begins. This happens when the temperature of the cylinder is zero degrees Celsius. If the cable is warmer the snowflakes melt. It is this melting that lowers the temperature of the conductor to $0^{o}C$. In the growth stage, the dominant thermodynamical processes are balanced to determine the rate of growth of the accretion. The third stage, collapse, occurs when the structural strength of the accretion is insufficient to hold it on the cylinder.

The first and second stages of the accretion process are governed by the rate of heat exchange per unit length of

transmission line. Terms include convection to the air stream (Q_v), evaporation or condensation (Q_e), melting (Q_m), and Joule heating (Q_J). In the first stage, the metal line also acts as a heat source. In the second stage, these four heat exchanges rates are balanced as follows:

$$Q_v + Q_e + Q_m + Q_J = 0.$$

The heat exchanges rates may be written as follows (Admirat et al., 1985a):

$$Q_v = K\ (T_a - T_s)\ \pi\ D_c,$$

$$Q_e = L_v\ \frac{Pr^{0.63}}{Sc^{0.63}}\ \frac{K\ (r_a - r_s)\ \pi\ D_c}{C_p},$$

$$Q_m = L_f\ c\ (1 - LWF)\ S\ PR\ \frac{(1 + U_c^2)}{W^2}\ D_c,$$

and $Q_J = R\ I^2$,

where

c = fraction of snow that must melt to maintain the heat balance,

C_p = specific heat capacity of air,

D = molecular diffusivity of water vapour in air,

D_c = diameter of the cylinder,

I = electrical current,

K = thermal conductivity of air,

L_f = specific latent heat of fusion,

L_v = specific latent heat of evaporation,

LWF = liquid water fraction of the accretion,

Pr = Prandtl number: the ratio of momentum diffusivity to thermal diffusivity,

r_a = mixing ratio of the air,

r_s = saturation mixing ratio,

R = electrical resistance per unit length of line,

S = sticking efficiency,

Sc = Schmidt number: the ratio of kinetic viscosity to molecular diffusivity,

T_a = temperature of the ambient air, in degrees Kelvin,

T_c = temperature of the transmission line in degrees Kelvin,

U_c = velocity component of the snowflake, perpendicular to the cylinder's axis,

W = fall speed of the snowflakes.

The equations for the heat exchange processes can be simplified by collecting the constants and expressing some of the variables in more convenient forms:

$$Q_v = 14.2 \; D_c^{0.61} \; U_c^{0.61} \; T_a,$$

$$Q_e = 24.3 \; D_c^{0.61} \; U_c^{0.61} \; [e_a(T_a) - e_s(T_c)],$$

$$Q_m = 0.93 \; c \; (1 - LWF) \; S \; PR \; \frac{(1 + U_c^2)}{W^2} \; D_c,$$

and $$Q_J = R \; I^2,$$

where the units of the heat exchange rates are Joules per second,

e_a = vapour pressure in the ambient air,

e_s = saturation vapour pressure over ice.

The model keeps track of the mass of the accretion and the amount of water present in the accretion. It calculates a radius for the accretion based on the mass of snow and its

assumed density. Due to the symmetry assumption, this type of model has the advantage of being quite simple. A hand calculator is all that is needed to make the calculations.

Mechanical models are much more complicated than thermodynamical models. They do not make the assumption of radial symmetry. In this aspect they provide a better representation of what happens in nature. An examination of the cross section of wet snow accretions (Wakahama et al., 1977) shows that the accretion builds asymmetrically on the windward side of the line, until the forces holding the accretion in place are overcome by gravity. When the accretion moves it rotates until its center of gravity is directly below the line. A series of layers caused by the rotations is often visible (see Figure 1) in cross sections of the accretions (Wakahama et al., 1977). A mechanical model (eg. Finstad, 1989; see section two of Appendix F) gives a better model of the shape and internal layering of wet snow accretions. This model also gives a reasonably accurate prediction of shape, mass, and density (Finstad, 1989).

The only time mechanical models use thermodynamics, if at all, is to calculate when accretion begins. A thermodynamic balance can estimate the liquid water content of the accretion. If the liquid water content of the accretion is not between 20% and 40% by mass, then it will have insufficient cohesive strength, and is likely to be blown off a transmission line (Wakahama et al., 1977).

Purely mechanical models cannot determine the fraction of the mass that is liquid. This is one of their important shortcomings. If thermodynamics is included in a mechanical model, then the accretion must be divided into layers and sections, and the thermodynamics of each section must be examined, as well as the thermodynamics of the interactions between adjacent section. As the accretion grows the size and shape of these sections changes. This makes modelling very difficult. Mechanical and thermodynamical process are not both used in the same model because difficulties in making and programming such a model.

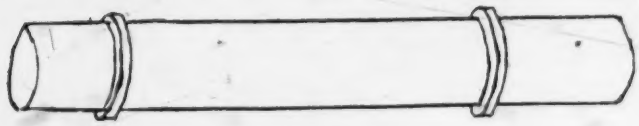
1.3.3 Prevention of wet snow accretions

Passive solutions, to the problem of preventing the growth of large wet snow accretions on transmission lines, have been proposed and even tested, but they have not proven to be effective for long line spans. Changes to the shape of the transmission line have been tested in Japan (Wakahama et al., 1977). Two shapes were tested by the Accretion Prevention Research Group of the Hokkaido Electric Company. One modification to the shape of the line was to place rings two to four millimetres thick around the line at intervals of 1.5 - 2 times the length of the stranding pitch (Figure 2a). This stopped accretions from sliding along the strands. However, this is only useful if the line has a great enough torsional rigidity and short enough span to prevent rotation of the line. This technique was effective for an ACSR (aluminum conductor steel reinforced) line with a diameter

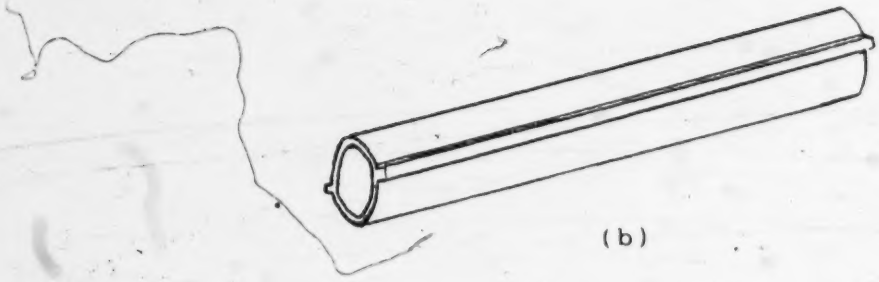
of 13.5 - 18 mm. For longer spans, 0.8 kg weights were attached to the line at intervals of less than 100m. When the rings and the weights were used together, they prevented the growth of large accretions of wet snow.

An alternative modification to the shape of wires was used for vinyl covered wires. Longitudinal fins were added to opposite sides of the wire (Figure 2b). These ran the length of the wire. They prevented the accretion from sliding around the vinyl. This technique also proved to be effective in urban situations were vinyl coated wires were used.

Currently there is no effective method to prevent long transmission lines from being downed during severe wet snow events. Moreover, severe wet snow events cannot currently be forecast with any precision. Thus, if they are to be prevented, they must be detected and dealt with on site. Given that the effects could be detected, and that personnel could reach the site of the problem in time, there are feasible means to prevent damage. The wet snow could be knocked or melted off the lines. However, in rural areas, detection is difficult, because severe wet snow events reduce both the visibility and the types of available transportation. While prevention through detection may be reasonable in some areas, in most areas it is highly impractical.



(a)



(b)

Figure 2 - Techniques of Accretion Prevention on Transmission Lines

The growth of large wet snow accretions on transmission lines can be prevented by modifying the shape of the lines. Two shapes have been found to be successful in preventing large accretions from growing on the short line spans typical of urban locations. Figure 2a shows rings two to four millimetres thick around the line at intervals of 1.5 - 2 times the length of the stranding pitch. Figure 2b shows fins added to opposite sides of the vinyl coated line.

THEORY OF WET SNOW ACCRETION

In order to thoroughly study the damage done by wet snow, the problem of accretion must be examined on several scales. The small-scale physics of the accretion process is examined to learn how the accretion begins to form, and how it grows. At the larger scale, the characteristics of precipitation events must be examined. This serves several purposes. One is to determine the meteorological conditions for the occurrence of wet snow. Another purpose, climatological in nature, is to determine the duration and the frequency of occurrence of potential wet snow events of a specific location. Potential wet snow events are precipitation events with a temperature between $-2^{o}C$ and $+6^{o}C$. These studies, used with an accretion model and a risk analysis, can be used to estimate the chance of occurrence and the severity of wet snow events.

A risk analysis (see Chapter 3) is the goal of this study, so only the maximum loads over a specific time period are of interest. The time period of interest, for our purposes, is one year. A risk analysis requires the extrema from at least thirty time periods (Gumbel, 1958). This means that measurements for at least thirty years are required. Since the maximum force on transmission lines is not routinely measured, the force must be estimated through the use of an accretion model (Finstad, 1989). The accretion model requires certain meteorological parameters (air temperature, relative humidity, precipitation rate, wind and

velocity) to calculate the force on the line. These meteorological parameters are not constant throughout a wet snow event, and some, such as wind velocity and precipitation rate, vary significantly with time. The horizontal force on the line is proportional to the square of the wind velocity, so the maximum force could occur at any time throughout the wet snow event. This means that the meteorological parameters have to be known (measured) at intervals, ideally before and after each change in wind speed, throughout the wet snow event. Meteorological observation stations are usually not near major transmission lines. Consequently there are very few measurements of the meteorological conditions during wet snow events, that were observed at the site of the transmission lines. To make up for this deficiency a model of potential wet snow events was devised.

2.1 The formation of wet snow in the atmosphere

Wet snow is made up of both liquid water and ice. For large accretions to adhere to transmission lines, between 20% and 40% (Wakahama et al., 1977) of the mass of the falling snow must be liquid water. Accretions can occur with liquid water fractions of the snowflakes of less than 20% if Joule heating is producing enough heat, relative to the precipitation rate (Admirat et al., 1985b). When less than 20% of the mass is liquid the snow retains some of the qualities of dry snow. It is not very sticky. If greater than 40% of the mass of most snowflakes is water then the

liquid water fraction of the accretion will eventually grow to be greater than 40%. When the liquid water fraction of the accretion is more than 40%, the structural strength of the accretion will decrease. This increases the likelihood of the accretion breaking off the line. The necessary fraction of liquid water limits the conditions under which wet snow accretions can occur. The falling snow must begin to melt, but cannot be excessively melted, by the time it reaches the ground. Matsuo and Sasyo (1981b) studied the melting process of wet snow. When the snow enters warm air there are three stages of melting. During all of the stages, the snowflake gains heat through convection from the ambient air. Initially, the heat loss due to sublimation, without melting, balances the heat gained by convection from the air. In the second stage, melting is required to balance the heat exchange; this is when the snow becomes wet. In the third stage the heat exchange requires rapid melting. Matsuo and Sasyo determined a theoretical condition for the formation of wet snow in a melting layer. Their paper contained only a graphical representation of the condition. The relationship between relative humidity and air temperature in the melting layer appears to be linear, and described by the equation:

$$RH \geq 100 - 12.5\ t_a, \qquad (1)$$

where the temperature is in degrees Celsius.

A simplified approach (Lozowski, Finstad, and Bourassa, 1989) similar to that of Matsuo and Sasyo can be used to

derive Equation (1). The heat exchanges taking place on the surface of a wet snowflake are assumed to be balanced. The effects of condensation are relatively small, and can be ignored to a first approximation. This leaves convection and sublimation as the balanced heat exchange processes:

$$F_K\ K\ (T_a - T_s) = F_D\ D\ L_s\ [\rho_s(T_s) - \rho_a(T_a)], \qquad (2)$$

where

D = molecular diffusivity of water vapour in air,

F_D = ventilation coefficient for diffusion,

F_k = ventilation coefficient for forced convection,

K = thermal conductivity of air,

L_s = latent heat of sublimation,

T_a = temperature of ambient air, in degrees Kelvin,

T_s = temperature of the surface of the snow flake, in degrees Kelvin,

ρ_a = vapour density of ambient air,

ρ_s = saturation vapour density of air.

The effects of ventilation in Equation (2) can also be ignored (Matsuo and Sasyo, 1981b), since the effect of ventilation on convection is approximately equal to the effect of ventilation on diffusion ($F_K = F_D$). Vapour density can be removed from Equation (2) through the use of the ideal gas law applied to the partial pressure of water vapour:

$$e = \rho R_v T, \qquad (3)$$

where

e = the partial pressure of water vapour,

R_v = specific gas constant of water vapour,

T = temperature in degrees Kelvin.

Substituting Equation (3) into Equation (2), assuming that the ventilation coefficients are equal, and rearranging, yields:

$$\frac{K R_v (T_a - T_s)}{D L_s} = \frac{e_s(T_s)}{T_s} - \frac{e_s}{T_a} \qquad (4)$$

Equation (4) is easier to evaluate when it is expressed in terms of the relative humidity (RH) rather than the partial vapour pressure of the ambient air:

$$e_a(T_a) = \frac{RH\ e_s(T_a)}{100}. \qquad (5)$$

Because the snowflake is melting, but is not completely melted, the temperature at the surface of the snowflake, T_a, is assumed to be 273 K. The minimum relative humidity (RH_c) for wet snow to occur may therefore be found from:

$$\frac{K R_v (T_a - 273)}{D L_s} = \frac{e_s(273)}{273} - \frac{RH_c\ e_s(T_a)}{100\ T_a}, \qquad (6)$$

where the temperature is in degrees Kelvin. If the relative humidity is less than this critical relative humidity (RH_c) then the snow will sublimate, and freeze or cool rather than melt. Note that the critical relative humidity refers to the relative humidity at some height in the atmosphere, rather than to the relative humidity at the surface. For melting to occur, the relative humidity must be greater than the critical relative humidity.

Equation (6) is non-linear. However, over the temperature range from 0°C to 5°C, the curve of the critical relative humidity as a function of temperature, described by Equation (6), approximates a straight line. A least squares fit to the curve yields the following linear approximation:

$$RH_c = 100 - 12.25\ t_a, \qquad (7)$$

where t_a is the temperature measured in degrees Celsius.

The maximum error in the critical relative humidity, due to approximating the relationship as linear, is a change of ±2.3%.

Equation (7) represents an approximate boundary condition, in a space described by relative humidity and air temperature, on the formation of wet snow. This condition must be met far enough above the surface for between twenty percent and forty percent of the mass of the snowflakes to melt. However surface conditions are the conditions which are most widely measured. Unfortunately, the Atmospheric Environment Service (AES) does not record the occurrence of wet snow. Therefore, in order to estimate the conditions needed for wet snow at the ground, the surface data for CFB Namao in Edmonton, Alberta were examined in relation to the condition of Equation (7), by Lozowski, Finstad, and Bourassa (1989). In twenty-one years of hourly data from April, 1966 to October, 1986 there were 1525 hours when precipitation (rain or snow) occurred with a temperature between -2°C and +6°C. These precipitation events will be

referred to as potential wet snow events. The range in temperature is wide enough to include all the wet snow events. A method similar to that used by Matsuo and Sasyo (1981c and 1981d) was used in the analysis of how surface values of relative humidity and air temperature affect the occurrence of wet snow. The possible existence of a relationship similar to Equation (7) was examined. The potential wet snow events were sorted, according to temperature, into bins of 0.25°C width. The minimum relative humidity for rain events was found for each bin (Figure 3). A least squares analysis was used to determine the best fitting line for the temperatures and relative humidities of these minima. Matsuo and Sasyo used only the bins which had snow present at relative humidities lower than the minimum relative humidities where rain occurred. This meant that the maximum relative humidity where snow occurred was also considered. Unfortunately there were insufficient data in our observations to allow for this extra condition: too many of the bins (21 of 32) did not have any snow events. The best fit line was:

$$RH_{cs} = 90.1 - 5.3\ t_a, \qquad (8)$$

where RH_{cs} = the surface critical relative humidity.

Despite the lack of snow events in some of the bins, Equation (8) is assumed to be a good discriminator between rain and snow, because Matsuo and Sasyo found similar equations to be good discriminators at three locations.

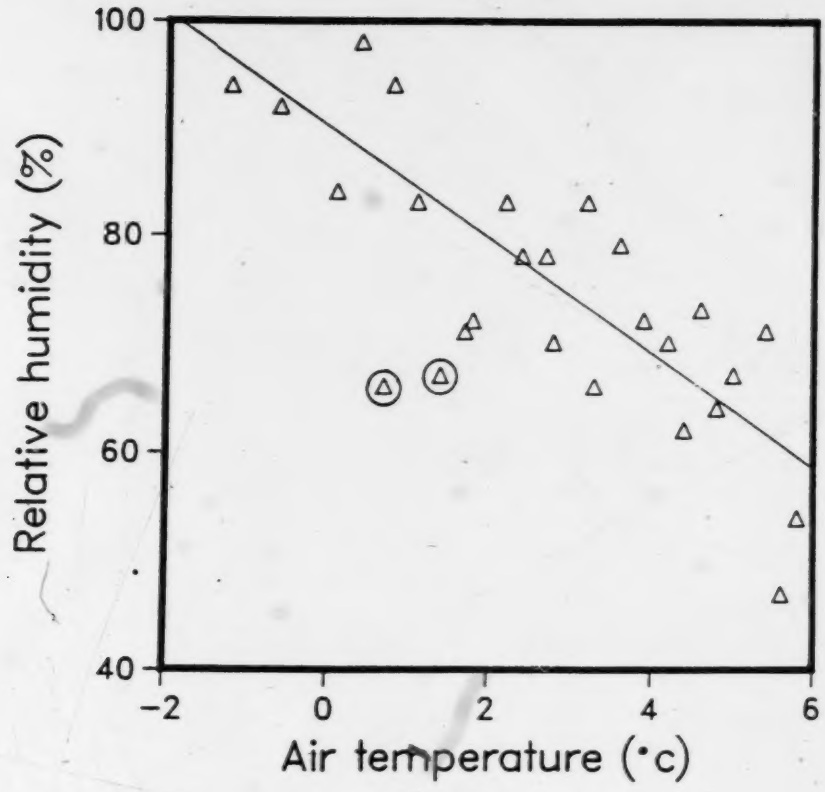


Figure 3 - Surface Critical Relative Humidity

Each triangle marks the lowest relative humidity at which rain was observed, within a temperature range. The range of each temperature bin was 0.25 ^{o}C. The two circled points were rejected as unreasonably extreme. They are observations from adjacent hours in a precipitation event. Equation (8) was determined from a linear regression for the 'best fit' line for the remaining points.

The meteorological processes represented by Equation (8) are probably similar to those describing Equation (7). If the surface relative humidity is less than the surface critical relative humidity, then the falling snow is probably dry. If the surface relative humidity is greater than the critical surface relative humidity, then there is probably a melting layer. This melting layer may be thick enough to cause snowflakes to melt entirely before they hit the surface. Because Equation (8) is a 'best fit' relationship derived from observations, it does not imply any theoretical limitation on the occurrence of wet snow. Equation (8) is a strong indicator of whether precipitation will be wet or dry, but it is not a completely accurate test. In Figure (3) there are two examples of rain occurring when the surface relative humidity was more than 30% less than the critical relative humidity.

Upper air data from Stony Plain, Alberta was used to attempt to confirm the relationships between the relative humidity and the temperature in Equations (7) and (8) (Lozowski, Finstad, and Bourassa, 1989). Stony Plain is Alberta's only regularly operating upper air station. Unfortunately the surface station (CFB Namao) and the upper air station are separated by 44 km, and Stony Plain is 85 metres higher than Namao. An examination of surface frontal analyses (by CMC) has shown that the presence of one or occasionally two fronts near the location of the potential wet snow event is not unlikely. This means that Namao will

sometimes be in a different air mass than Stony Plain. Consequently the conditions at Namao may not be indicative of the conditions at Stony Plain. Nevertheless, it will be assumed that the surface conditions at Namao are representive of the surface conditions at Stony Plain.

The upper air soundings at Stony Plain are taken only every twelve hours, which is relatively infrequent compared to the hourly surface data. Ninety-three upper air soundings were observed during potential wet snow events. Some soundings were observed during the same surface event, so of the 163 surface events approximately 90 are used to test the validity of Equations (7) and (8). An examination was made of sounding profiles likely to produce wet snow or rain (results are in Appendix B). These were assumed to be the profiles with a significantly thick layer (1 kPa) of air with a relative humidity in the layer greater than the critical relative humidity (RH_c, equation 7) needed for wet snow. The layer thickness of one kiloPascal (approximately 100 m) was used to insure that the falling snow was well into the melting layer before reached the ground. Seven soundings had sufficient relative humidity, but insufficient thickness. Fifty-three of the soundings met both conditions. Of these fifty-three soundings, only three had a melting layer which did not extend to the ground. Reversing the direction of the argument leads to the conclusion that when the surface relative humidity is greater than the critical relative humidity, RH_c, there is likely to be a melting

layer sufficiently large for falling snow to become wet snow (or rain). Thus the surface conditions are likely to be a good indication of the relevant upper air conditions.

To investigate the practical applications of Equations (7) and (8), a scatter plot (Figure 4), with axes of relative humidity and temperature, was made of the sixty-two soundings corresponding to snowfall at Namao. The thirty-one with a sufficiently thick layer of air with a relative humidity greater than that of the critical relative humidity (RH_c), were assumed to produce wet snow. The other thirty-one precipitation events were assumed to produce dry snow. Both lines describing the critical relative humidity and the surface critical relative humidity are shown on the plot. Twenty-nine of the thirty-one precipitation events, that met the criterion for wet snow given on page twenty-six, had surface relative humidities exceeding the surface critical relative humidity. Only two of the wet snow events lie below this line (see Figure 4 and Table 1). Thus the surface critical relative humidity seems to provide a good approximate boundary for the occurrence of dry snow. To a good approximation, if the relative humidity at the surface is less than the critical surface relative humidity then, the snow should be dry.

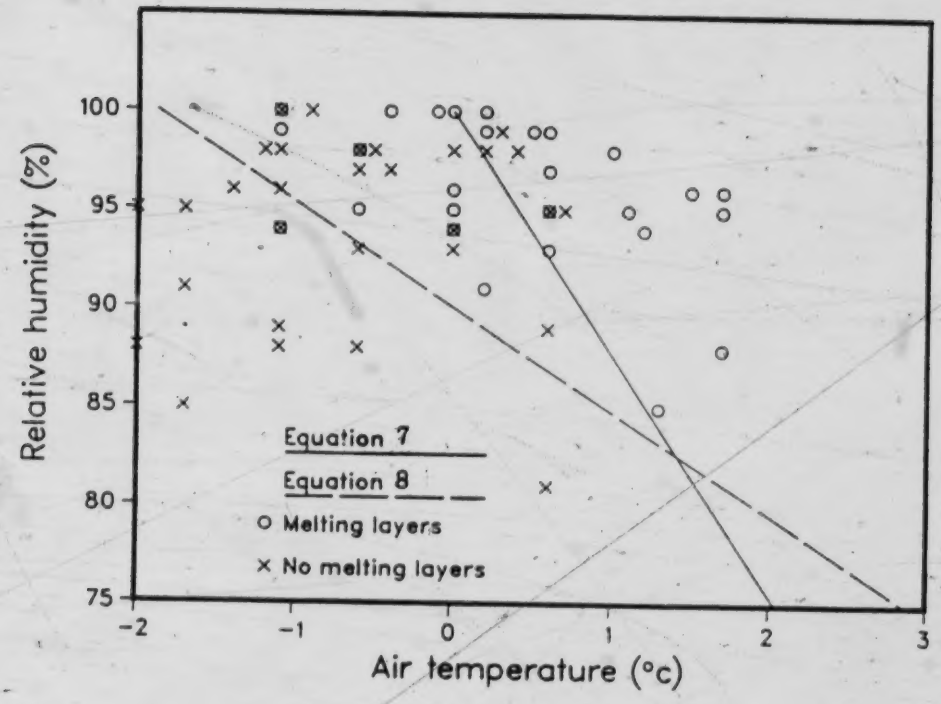


Figure 4 - Relative Humidity Regimes

Surface relative humidity versus surface air temperature at CFB Namao, for times at which upper air soundings exist and falling snow was observed. The symbols indicate whether or not there was a sufficiently thick (1 kPa) melting layer over Stony Plain. The critical relative humidity (RH_c), Equation (7), is represented by the solid line, and the surface critical relative humidity (RH_{cs}), Equation (8), is represented by the dashed line.

Table 1 - Relative Humidity Regimes

	$RH < RH_{cs}$	$RH_{cs} < RH < RH_c$	$RH > RH_c$	Total
Wet Snow	2	9	20	31
Dry Snow	14	11	6	31
Total	16	20	26	62

The critical relative humidity line also makes a good boundary. Twenty of the twenty-six soundings with relative humidities greater than the critical relative humidity met the criterion for wet snow given on page twenty-six. In a model it would be reasonable to set the chance of wet snow occurring, given that the relative humidity is greater than the critical relative humidity, to be 77% (20/26 = 0.769). However, the six dry snow events with relative humidities greater than the critical relative humidity are all close to the critical relative humidity. It is possible that they lie above the critical relative humidity line because of differences between the surface conditions at Namao and the surface conditions at Stony Plain. If a front lay between the sites, it could cause changes in temperature large enough to move the points from positions below the critical relative humidity to the positions of the six dry snow points. If the object of the model is to determine a worst case analysis or a risk analysis, it would be prudent to assume all events, with surface conditions in this regime, are wet snow events. In the extreme accretion model this is assumed.

In the region where $RH_{cs} < RH < RH_c$ nine (45%) of the twenty points met the criterion for wet snow given on page twenty-six. Dr Finstad (Lozowski, Finstad, and Bourassa, 1989) added the requirement that events in this regime, in order to be considered wet snow events, should have a surface temperature greater than $1^{o}C$. This is a typographical error, since in this regime there are no points with an air temperature greater than $1^{o}C$. It should read $-1^{o}C$. In this region there is only one point with an air temperature less than $-1^{o}C$. One observation is insufficient to reasonably assume any restriction. Further, because relative humidity cannot exceed 100%, the minimum temperature, where it is possible for the relative humidity to exceed the critical relative humidity, is $-1.87^{o}C$. The region between $-1^{o}C$ and $-1.87^{o}C$ is small enough to make the added restriction almost meaningless. However, wet snow will not occur on the ground with temperatures below zero degrees Celsius, unless there is a melting layer aloft. Then the wet flakes could develop within the warm layer aloft before they pass through the cold layer near the ground. The model of annual extreme accretions treats the occurrence of wet snow at a surface temperature less than $0^{o}C$, as though the snow fell through a warm inversion layer. When the inversion occurs the average change in temperature between the warmest height in the inversion and the surface is $2.3^{o}C$ (Lozowski, Finstad, and Bourassa, 1989). Given the current data, in the regime where $RH_{cs} < RH < RH_c$, the chance of the occurrence

of wet snow is 45%. The accuracy of the chance of occurrence of wet snow in this regime would probably be improved if the upper air station and the ground station were closer together.

Using surface data alone (RH and t_a), it is possible to estimate the chance of a wet snow event occurring provided that the snow occurs. These chances are:

$RH > RH_c$	100% chance of wet snow occurring,
$RH_{cs} < RH < RH_c$	45% chance of wet snow occurring,
$RH < RH_{cs}$	no chance of wet snow occurring.

There are several problems involved with using this set of criteria for forecasting wet snow events. One problem is the inaccuracy of the predicted time of the wet snow event. Forecasts of temperature and relative humidity are issued at six hour intervals, not in hourly weather updates. The six hour time step means that it will be very difficult to accurately estimate the time that the wet snow event begins, and the duration of the event. It is possible to program a wet snow forecasting routine into a weather forecasting model. Then the occurrence of wet snow could be estimated for each time step of the model. Since these time sets are usually less than one hour, the time of the onset of the event, and the duration of the event could be more accurately estimated to within an hour.

The second problem in forecasting the occurrence of wet snow events, is the large influence of small errors in relative humidity and temperature on the chance of

occurrence. An error of 0.4 ^{o}C in temperature causes an error of 5% in the critical relative humidity, and an error of 2.1% in the surface critical relative humidity. Considering that the relative humidity is usually over 90% during potential wet snow events these are large errors. These errors can significantly effect the prediction of the chance of the occurrence of wet snow. A third problem is that it is difficult to accurately predict the time, location, and intensity of precipitation events. Forecasts of wet snow accretions, based an visual examinations of the AES surface meteorological predictions, are inadequate. An accretion model added to a weather forecasting model would be more accurate than a forecast produced by a visual inspection. However, the weather forecasting model would have to accurately predict the temperature to within a approximately half a degree before it would be useful in forecasting the loads due to extreme wet snow accretions.

2.2 The accretion of wet snow on transmission lines

The accretion of wet snow on a transmission line can be treated as an accretion on a circular cylinder (Finstad, 1989; Admirat et al, 1985a). The number of snowflakes that hitting the cylinder depends on the rate of precipitation, the cross sectional area of the cylinder perpendicular to the direction in which the snowflakes move, and the parameter of the air flow around the cylinder: the Reynolds number. The ratio of the number of particles that hit the cylinder, to the number that would hit the cylinder if it

did not disturb the air flow, is referred to as the collision efficiency. Since the air is forced to move around the cylinder, some of the snowflakes that would otherwise hit the cylinder are transported by the air flow around the cylinder. Thus the collision efficiency is less than or equal to unity. The calculation of the collision efficiency can be further complicated by turbulence. There may be a turbulent wake behind the cylinder. As snow accretions grow, they become non-circular and rough (see Figure 5). The increased roughness increases the size of the turbulent wake. Because of the turbulence, flakes that would have passed close by the cylinder can be drawn in and collide with the back of the cylinder. Little is known about how this complicating factor effects snowflake trajectories. Research (Wakahama et al., 1977; Admirat et al., 1985b) has shown that snowflake trajectories, prior to hitting the cylinder, can be very accurately approximated as straight lines. Neither studies detailed an investigation of the minimum velocity needed for this assumption to be true. The minimum velocity used in the study be Admirat et al. was 5 m/s, so the minimum velocity must be equal or less than 5 m/s. This means that the complications can be ignored, and the collision efficiency for wet snowflakes can be approximated as unity. However, not all snowflakes that hit the cylinder (or the accretion) will stick to the surface. They may splash or bounce (Wakahama et al., 1977). The ratio of the mass of the snowflakes that stick to the cylinder, to

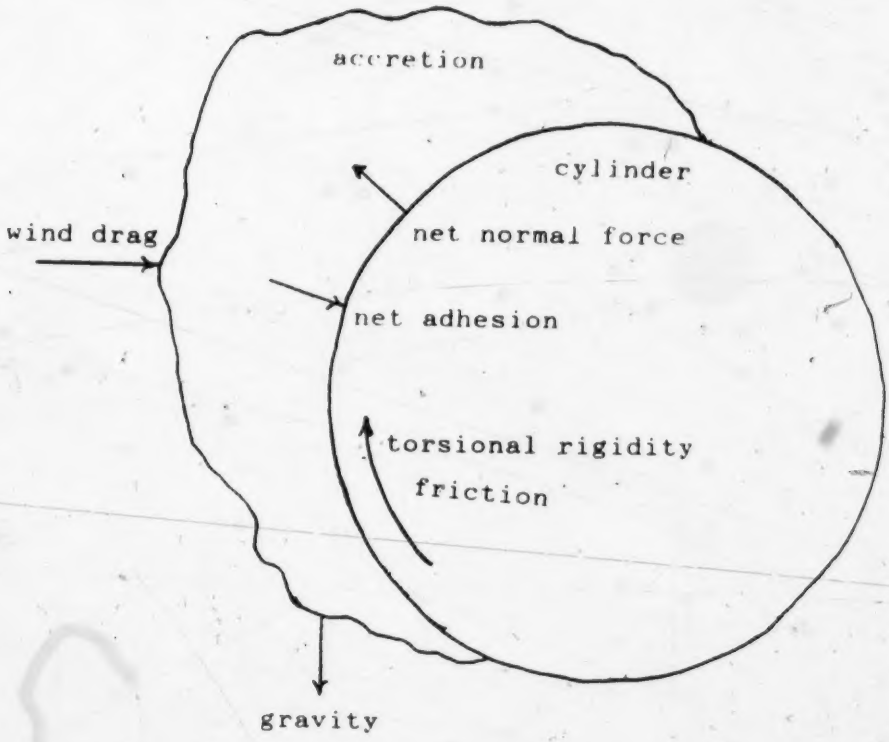


Figure 5 - Forces Acting on the Accretion

There are six external forces that act on the accretion. Friction, torsional stiffness, and the normal force of the cylinder on the accretion are all resistive forces. The net normal force of the cylinder on the accretion is equal and opposite the net force the accretion exerts on the cylinder. The accretion will rotate only if the net forces on the accretion are unbalanced.

the mass of the snowflakes that hit the cylinder, is called the sticking efficiency. The product of the collision efficiency and the sticking efficiency is called the collection efficiency. The sticking efficiency is likely to be influenced by the momentum of the snowflake, its liquid water content, and its shape. An equation for the sticking efficiency has been developed by Finstad and Lozowski (1989) based on the existing experimental work (Wakahama et al., 1977; Admirat et al., 1985b). This is:

$$S = \frac{0.038\ t_a}{U_c\ D_c}, \text{ or } S = 1, \text{ whichever is less,} \quad (9)$$

valid for: $0 < t_a < 4^{o}C$,

$5 < U_c < 15$ m/s, where U_c is the speed of the snowflake perpendicular to the axis of the cylinder,

$0.01 < D_c < 0.4$ m, where D_c is the diameter of the cylinder including the accretion.

As the accretion grows around the cylinder, the diameter D_c increases. However, wet snow accretions are not cylindrical (Wakahama et al., 1977). The change in shape will therefore decrease the quality of this approximation. This approximation is nevertheless used in the annual extreme accretion model.

The snow approaches the line from the direction of the sum of the wind and fall velocity vectors. It moves in approximately straight trajectories, and if it sticks to the line, it stays where it hits the line. Consequently growth occurs only on the windward side of the line, and the

accretion growth is not radially symmetrical (Wakahama et al., 1977) as is assumed in current thermodynamical models (Admirat et al., 1985a). The accretion grows until the center of mass of the accretion is horizontally displaced far enough away from the cylinder (the center of rotation) that the torque due to gravity is great enough that there is a net force on the accretion. In other words the torques due to gravity and adhesion exceed the torques due to wind drag, torsional stiffness, and the normal force of the line on the cylinder (see Figure 5). If the surface is smooth, the accretion can then either slide down around the cylinder if static friction is overcome, or it can rotate with the line if static friction is not overcome. Transmission lines are usually not smooth, but stranded. They have little torsional rigidity, and observations have shown that lines do rotate with the accretion. In nature, and in Finstad's accretion model, the center of the line may make several complete rotations (Finstad, 1986). This growth process continues until the conditions for growth are no longer met; either the precipitation ends, it changes to a form other than sticky, wet snow, the accretion falls off, or the line falls down.

Little is known about how wet snow accretions fall off transmission lines. In the early stages of growth, they are susceptible to being blown off the line by gusts of wind (Wakahama et al., 1977). Gusts are already considered (Lozowski et al., 1989) in determining the maximum force on

a line. Through further research it might be possible to add an additional gust-related growth limiting factor for the beginning of the accretion process. The structural strength of wet snow is also unknown. If it is very wet or very dry, it will lack structural strength and fall off the line. Evidence (Wakahama et al., 1977) suggests that the liquid water moves towards the center of the accretion. The physics of the structural strength of wet snow and the movement of water in the wet snow are largely unknown. Research into these areas might make it possible to model the structural limiting factors to the accretion process.

2.3 Modelling theory

Numerical Models are mathematical representations of processes and the interaction among these processes. The ideal model includes the physics of every process related to the problem of interest. It does not use statistics except were randomness is a natural part of the phenomena (e.g. quantum mechanical tunnelling). Ideal models are rarely developed. Usually the physics of at least one of the processes involved is not understood. Processes that are not understood must be approximated from an extrapolation of experimental results. Another major problem with models is the time required to use the model to get a particular result. A model that considered all the processes related to the problem would usually take far to long to generate the desired result. To reduce the computational time, an examination of the influence of each process is made, and

those with the least influence are ignored, while other processes are approximated. In practice these approximations are also influenced by the difficulty in modelling a particular process. Those that are difficult to model are often approximated. Wet snow models require approximations to be made for all these reasons. For example, the assumptions made in determining the collection efficiency were described in the previous section. Two additional assumptions are that the transmission line is straight, and that the physical characteristics of the accretion are uniform along the line.

There are insufficient data to perform a proper risk analysis on wet snow events. There should be at least thirty time periods (years in this case) of data (Gumbel, 1958). A risk analysis uses the maxima of the phenomenon being considered (the mass of the accretion, vertical force on the line, and horizontal force on the line) for each year. Records of the size and mass of wet snow accretions have been kept only in the last few years. These are only of limited value as the accretions are usually not examined until a day or two after the event. However, given suitable environmental input, a good model could simulate the size and mass of wet snow accretions on a transmission line. A risk analysis could then be performed using the data generated by the model. The problem with this approach is the accuracy of the input parameters for the model. Meteorological recording stations are usually not situated

close enough to major transmission lines to provide accurate meteorological data as input for the model. This means that the input parameters also have to be simulated by a model.

The model of the input parameters for the wet snow accretion model is a time series for each of the input parameters. In other words it is a model of how these parameters develop with time. The parameters of interest are air temperature, relative humidity, precipitation rate, and wind velocity (speed and direction). This model should consider any time dependent trends, and produce values with the same means and standard deviations as the observed values. It should also consider the correlations between the parameters, the correlations between the changes in the various parameters, as well as the correlations between a particular parameter and the change in the same parameter. A probability distribution of the initial values of the parameters is also necessary. The model of the input parameters must have similar statistics to a sample set of data collected under conditions similar to those in which the model would be used (potential wet snow events).

The model that was developed for the meteorological input parameters required by Finstad's wet snow accretion model was a model of potential wet snow events rather than for actual wet snow events. This is more practical than modelling actual wet snow events because precipitation events can easily shift from wet snow to either rain or dry snow, and back again. Modelling potential wet snow events

eliminates the difficult constraint that the precipitation must be wet snow. In the model for the meteorological parameters, relative humidity and air temperature are examined to determine if the conditions for wet snow exist. If the conditions for wet snow exist, then the wet snow accretion model is used. Since the meteorological variables are not modelled for all times and conditions, it is necessary to model both the duration of potential wet snow events, and the annual number of potential wet snow events. These considerations will provide an adequate model of the input parameters for the wet snow accretion model. The annual maximum wet snow accretions, needed for the risk analysis, can then be found by modelling all the potential wet snow events in a year.

THEORY OF RISK ANALYSIS

Risk analysis is used to estimate the chance (risk) of something happening. It can be used to determine the quality of construction needed for a structure to have a particular chance of lasting a specified time. Risk analysis also can be used to estimate the frequency of repairs. Given estimates of the costs of construction and the likely costs and frequency of repairs, it is possible to estimate the most cost effective quality of construction.

Repairs are needed when the force on a transmission line exceeds the tolerance or failure strength of the line or the towers. The average time between structural failures can be estimated as a mean return time. The mean return time (T), of the occurrence of an event of magnitude equal to or greater than a specific magnitude, is the average time between these events. If p is the probability, in a specific time, that the force on the line will exceed the force it can tolerate, then the mean return time T is equal to 1/p.

That is, if $p = P(x > x_o)$,

then $T(x_o) = 1/p.$ (10)

where x is the force on the line,

and x_o is the maximum force that the line can sustain without damage.

A proof of this statement is given in Appendix A. The units of time for the return period are equal to the time period over which the probability applies. For example, if the

probability is for a ten minute period, then the units of the return period are tens of minutes. In the case of an annual analysis of extremes, the most reasonable unit for time is years.

The risk (r) is defined as the probability that one or more events will occur, that exceed the system's tolerance, during a specified time interval (t). The risk can be determined as follows (a proof is given in appendix A):

$$r = 1 - (1 - p)^t. \tag{11}$$

Equation (10) can be used to replace the probability of exceedance with mean return period:

$$r = 1 - (1 - T^{-1})^t. \tag{12}$$

The same units of time must be used for the time interval, and the return period; one unit of time must be equal to the interval over which the probability applies. In practical applications, the time interval, t, is the desired life time of the system. Table 2 displays risk as a function of the probability and the time interval. Note that even for a low probability such as 0.02 (an occurrence of once in every fifty time units on the average) the risk rises rapidly with the length of the time interval.

Table 2 - Risks

Time	Probability of Occurrence						
Int.	0.50	0.30	0.10	0.05	0.02	0.01	0.001
2	0.7500	0.5100	0.1900	0.0975	0.0396	0.0199	0.0020
5	0.9688	0.8319	0.4095	0.2262	0.0961	0.0490	0.0050
10	0.9990	0.9718	0.6513	0.4013	0.1829	0.0956	0.0100
20	1.0000	0.9992	0.8784	0.6415	0.3324	0.1821	0.0198
30	1.0000	1.0000	0.9576	0.7854	0.4545	0.2603	0.0296
40	1.0000	1.0000	0.9852	0.8715	0.5543	0.3310	0.0392
50	1.0000	1.0000	0.9948	0.9231	0.6358	0.3950	0.0488
75	1.0000	1.0000	0.9996	0.9787	0.7802	0.5294	0.0723
100	1.0000	1.0000	1.0000	0.9941	0.8674	0.6340	0.0952
250	1.0000	1.0000	1.0000	1.0000	0.9936	0.9189	0.2213
500	1.0000	1.0000	1.0000	1.0000	1.0000	0.9934	0.3936

It is possible to express the mean return period as a function of the risk and the time interval. To do this we add one to both sides of Equation (12) and take the t^{th} root:

$$1 - T^{-1} = (1 - r)^{1/t}. \qquad (13)$$

Rearrange Equation (13) to isolate the mean return period gives:

$$T^{-1} = 1 - (1 - r)^{1/t} = p. \qquad (14)$$

Finally inverting both sides of Equation (14) yields:

$$T = [1 - (1 - r)^{1/t}]^{-1}. \qquad (15)$$

This is useful because architects have specific maximum risks and minimum life times (time intervals) in mind when

they design a structure. With Equation (15) it is possible to determine the length of the mean return period associated with the desired risk and the designed life time. The mean return period and probability are associated with the magnitude of the phenomenon that is creating the risk. This in turn, gives the architect an approximation of the forces his structure will have to be able to withstand. Knowing this he can attempt to design the structure to withstand these forces.

The following table (Gumbel, 1958) gives an indication of how great a tolerance must be built into a system for it to last a long time with little chance of failure. It lists the number of mean return periods, for events which an object must be constructed to withstand, to have a specified chance of failure over a specified time.

Table 3 - Minimum Designed Mean Return Periods

Risk	Time Interval t							
r%	2	5	10	15	20	25	50	100
75	2.00	4.02	6.69	11.0	14.9	18.0	35.6	72.7
50	3.43	7.74	14.9	22.1	29.4	36.6	72.6	145
40	4.44	10.3	20.1	29.9	39.7	49.5	98.4	196
30	6.12	14.5	28.5	42.6	56.5	70.6	141	281
25	7.46	17.9	35.3	52.6	70.0	87.4	174	348
20	9.47	22.9	45.3	67.7	90.1	113	225	449
15	12.8	31.3	62.0	90.8	124	154	308	616
10	19.5	48.1	95.4	143	190	238	475	950
5	39.5	98.0	196	293	390	448	976	1949
2	99.5	248	496	743	990	1238	2475	4950
1	198.4	498	996	1492	1992	2448	4975	9953

For annual wet snow extremes the appropriate units of time are years. To have a 10% risk of a power outage, over ten years, the system would have to be built to withstand the effects of an event that occurs on average once in 95.4 years! The problem of design standards is also affected by the standards of society. The maximum socially acceptable risk will set the minimum acceptable mean return period (tolerance) for which a system should be designed.

3.1 Extreme Value Analysis

Ideally a relationship can be found between the magnitude of the phenomenon creating the risk and either the probability of exceedance or the mean return period. Extreme value analysis can be used to find this relationship. Gumbel (1958) has described the use and purpose of extreme value analysis in the following manner: "the statistical theory of extreme values deals with the behavior of the largest observations in a statistical series and serves for the forecast of extremes". It is the ability to forecast the extremes that makes it possible to find the relationship between the extremes of a phenomenon and the mean return period or the exceedance probability related to these extremes.

The extreme values of interest, the horizontal load and the vertical load, have no upper limit to their magnitude. The lower limit is zero. In other words they are unbounded at the upper end, and bounded at the lower end. This means they are likely to fit what is called an exponential probability distribution (Kinnison, 1985):

$$P(x_o > x) = \exp(-\exp(-Y)), \qquad (16)$$

where $Y = (x - u) / a,$ (17)

and u and a are constants.

'Y' is called the reduced variate. If the distribution of extreme values is an exponential distribution then Equations (16) and (17) can be used to relate the probability of

occurrence to the magnitude of the extreme value. To do this the constants 'a' and 'u' must be determined.

There are several methods that can be used to determine the constants of the distribution 'a' and 'u'. The two methods that will be used will be referred to as the method of moments and the regression method (Kinnison, 1985). The method of moments uses the constant's similarity, in Equation (17), to a standard deviation and a mean. While 'a' is not a standard deviation it is a measure of the dispersion of the extremes. Similarly 'u' is not the mean of the extremes, but it is related to the mean. Let ME. be the mean of the extreme values, and let SDE be the standard deviation of the extreme values. Then,

$$SDE = \pi a/6^{0.5}, \tag{18}$$

and $$ME = u + a\gamma, \tag{19}$$

where γ is Euler's constant: $\gamma = 0.5772....$

These can be solved for 'a' and for 'u':

$$a = SDE\ 6^{0.5}/\pi \tag{20}$$

$$u = ME - a\gamma. \tag{21}$$

Kinnison implies that this method may be more accurate than the regression method.

The regression method determines the constants 'a' and 'u' by finding the best fit line for the relationship between the reduced variate and the extremes. Equation (17) shows that if the distribution of extremes is an exponential distribution then this relationship will be linear. In practice a variation of Equation (17) is used:

$$x = u + a\,Y. \qquad (22)$$

The slope of the extremes as a function of the reduced variate is equal to 'a'. The y-intercept (Y = 0) of Equation (18) is equal to 'u'. The reduced variate is estimated by estimating the probability of exceedance, and by using the functional inverse of Equation (16):

$$Y = -\ln(-\ln(1 - p)). \qquad (23)$$

In order to estimate the exceedance probability, the extremes must be ordered from smallest to largest. If there are N, extreme values then the probability of non-exceedance of the m^{th} extreme can be approximated as a percentile:

$$P(x \leq x[m]) = m / (N + 1). \qquad (24)$$

The probability of exceedance is one minus the probability of non-exceedance:

$$P(x > x[m]) = 1 - m / (N + 1). \qquad (25)$$

Equation (25) can be substituted into Equation (23) to provide an estimate of the reduced variate for each extreme:

$$Y = -\ln(-\ln[m / (N - 1)]). \qquad (26)$$

To use the regression method, there must be enough extreme values to give a reasonable estimate of the reduced variate. Gumbel (1958) recommends at least thirty values.

If the constants 'a' and 'u' can be found, it is possible to determine a relationship between the extreme values and the exceedance probabilities. This is achieved by substituting Equation (23) into Equation (22):

$$x = u - a \ln[-\ln(1 - p)]. \qquad (27)$$

It should be recalled from page 46 that the methods for determining the constants 'a' and 'u' depend upon the assumption that the distribution of the extremes is an exponential distribution. This assumption can easily be tested when the regression method is used to determine 'a' and 'u'. The coefficient of regression is a measure of the 'goodness of fit' of the linear relationship. If the fit is good, then the assumption about the distribution of extremes is likely to be true.

RESULTS AND DISCUSSION OF SURFACE AND RISK ANALYSES

Twenty-one years of hourly surface meteorological data were examined to produce a model of annual potential wet snow events. The data came from CFB Namao from April, 1966 to October, 1986. The meteorological readings were recorded every hour. The data set was reduced to potential wet snow events by chosing only the times with precipitation and a temperature between $-2^{o}C$ and $+6^{o}C$. This left 1525 hours of data. The data for every hour of each variable was sorted according to the number of hours from the onset of the precipitation event. The hourly means of some of the variables varied as a function of time. This meant that a criterion had to be established to determine the beginning and the end of a precipitation event.

Obviously events begin with the onset of precipitation. Initially, the end of an event was defined as the end of the last hour, in one or more concurrent hours, when precipitation occurred. This did not seem entirely reasonable as a single hour could therefore separate two events. For such cases one long event seemed to be an equally valid representation. It would also be a more accurate representation for calculating the effects of wet snow accretion on transmission lines. In nature, a lull of one or two hours in the precipitation would have little effect on the final size of the accretion, provided that there was little or no melting. Thus statistics that treat two events that are nearly adjacent in time as one long

event would appear to provide a better representation of the meteorological parameters required for modelling the accretion process. In order to allow for this possibility, a lull of one hour without precipitation was not considered to end an event, provided this hour occurred after the first three hours of the event. Similarly, a lull of two hours would not end an event, provided it occurred after the first five hours. Using this criterion, there where 464 potential wet snow events with 154 events having durations of three or more hours.

4.1 Results of the analysis of surface input parameters

The model for wet snow accretion used in the extreme value model requires some surface parameters as input. These parameters are: air temperature (T_a), relative humidity (RH), wind direction (dir), wind speed (U), and rate of precipitation (PR). The extreme value model also needs the duration (dur) of the event, the number of potential wet snow events in a year, and the number of years to be simulated. The number of years is arbitrary and can be set interactively when the program is run. The frequency distribution of the number of potential wet snow events in a year made a good fit when compared to a Gaussian distribution (see Figure 6). Only the nineteen complete years were used to determine the statistics of the annual number of potential wet snow events. The distribution had

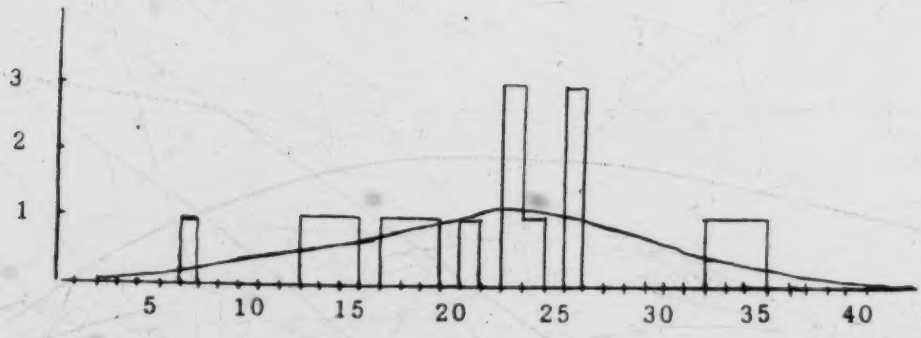


Figure 6 - Distribution of the Number of Annual Potential Wet Snow Events

A Gaussian curve is superimposed on the histogram of the number of potential wet snow events in one year. The mean number of events is 22.632 events per year. The standard deviation is 7.9 events per year.

the following statistics:

mean: 22.632 events per year

standard deviation: 7.946 events per year

reduced chi squared: 0.683.

This reduced chi squared value (Taylor, 1982) indicates that there is a 40% chance that the hypothesis that the distribution is not Gaussian should be rejected. This means that a Gaussian distribution is a fairly good approximation. Therefore the distribution of the number of events in a year was simulated by a Gaussian distribution.

A very good distribution also exists for the duration of potential wet snow events. A linear relationship exists between the log of the duration and the log of the number of events with that duration (see Figure 7):

$$\log(\text{dur}) = m \log(\text{number of events with this dur}) + b \quad (28)$$

A least squares analysis (Taylor, 1982) was used to verify this. The results were:

slope, m:	-1.577
standard deviation of the slope:	0.128
Y intercept, b:	2.290
standard deviation of the Y-intercept:	0.139
correlation coefficient:	-0.937
data points:	23

were X is the log (base 10) of duration (hours), and Y is the log of the number of events of a given duration. Equation (28) can be used to determine a relationship between the duration (dur_o) and the probability that the

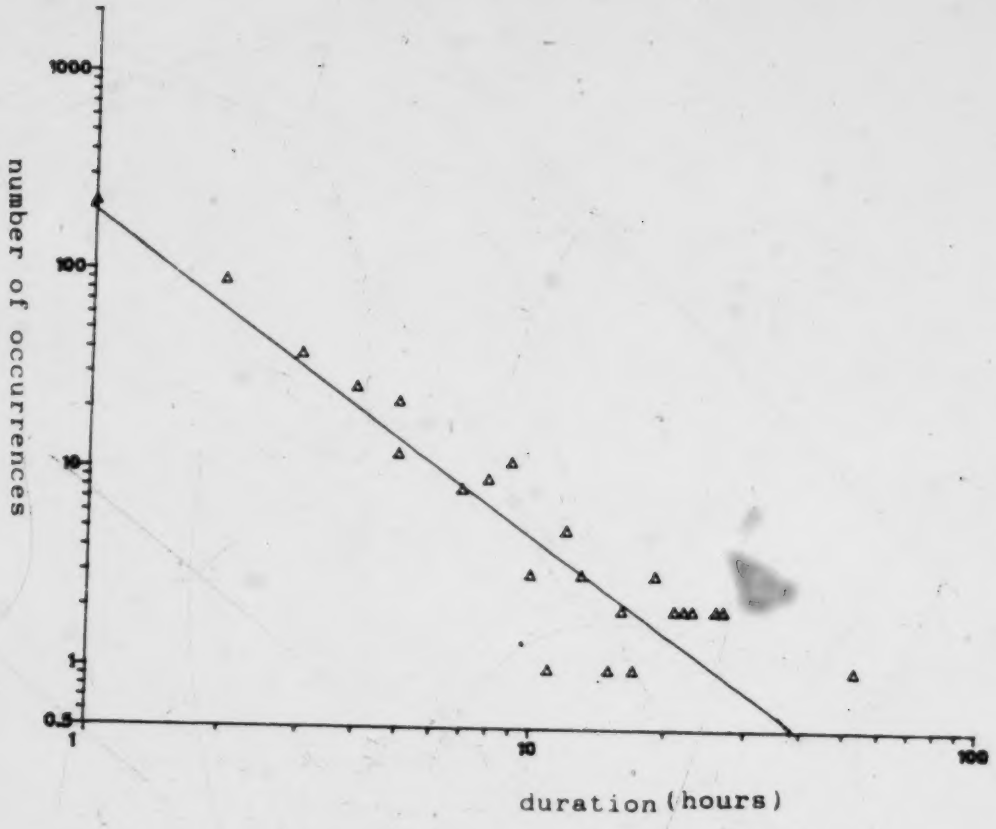


Figure 7 - Distribution of the Duration of Potential Wet Snow Events

The vertical axis is the log of the number of potential wet snow events for a given duration, and the horizontal axis is the log of the duration of potential wet snow events. The distribution makes a good fit (correlation coefficient of -0.937) to a linear relationship. The 'best fit' line is shown.

duration of an event will be equal to or less than that duration:

$$dur_o = f(P(dur_o \leq dur)) = f(dur). \quad (29)$$

In other words it is possible to construct a probability function for the duration of a potential wet snow event. To demonstrate this let

X' = log(dur), a dummy variable for X,

X = log(dur_o),

Y = log(number of events),

$X_o = X'(Y=0) = -b/m$,

m = slope of Equation (28),

b = Y-intercept of Equation (28),

$P = P(X \leq X') = P(dur_o \leq dur)$.

From the least squares analysis it is apparent that Y is (to a close approximation) a linear function of X':

$$Y = m X' + b. \quad (30)$$

By definition P is the integral of Y from $X'=0$ to $X'=X$, divided by the integral of Y from $X'=0$ to $X'=X_o$. Thus:

$$P = \frac{0.5\ m\ X^2 + b\ X}{0.5\ m\ X_o^2 + b\ X_o}. \quad (31)$$

Equation (31) may be rearranged as follows:

$$0.5\ m\ X^2 + b\ X - (0.5\ m\ X_o^2 + b\ X_o)\ P = 0. \quad (32)$$

Rewriting X_o in terms of the slope and y-intercept:

$$X_o = -b / m. \quad (33)$$

This was substituted into Equation (32), and Equation (32) was multiplied by 2/m to yield:

$$X^2 + \frac{2\ b}{m} X + \frac{b^2}{m^2} P = 0 \quad (34)$$

This is a quadratic equation so X can be solved for using the quadratic formula:

$$X = [(1 - P^2)^{0.5} - 1] b / m. \quad (35)$$

The antilog of this expression was used to express duration in terms of the cumulative probability of the occurrence of that duration (see Figure 8):

$$dur_o = \text{Antilog}([(1 - P^2)^{0.5} - 1] b / m). \quad (36)$$

Both the duration of events and the number of events in a year are therefore easy to generate in a model. To generate the duration, P is replaced by a random number between zero and one. Assuming that the random number generator has a uniform distribution, this will produce a very accurate simulation of the distribution of the duration of an event. Its shortcoming is that it limits the duration to twenty-eight ($10^{-b/m}$) hours. Durations of greater than twenty-eight hours are possible in nature, but they are unlikely. Only one event of the 464 observed potential wet snow events had a duration greater than twenty-eight hours. In practice it is impractical to model events of greater than twenty-eight hours in duration.

Two analyses were required to simulate the behavior of each meteorological parameter: an examination of its initial value, and an examination of how the parameter changes with time. The distribution of initial conditions (see Figures 9 to 11) for each parameter can be approximated as Gaussian. However, this is not an accurate approximation. Chi squared tests indicated a probability of less than one percent that

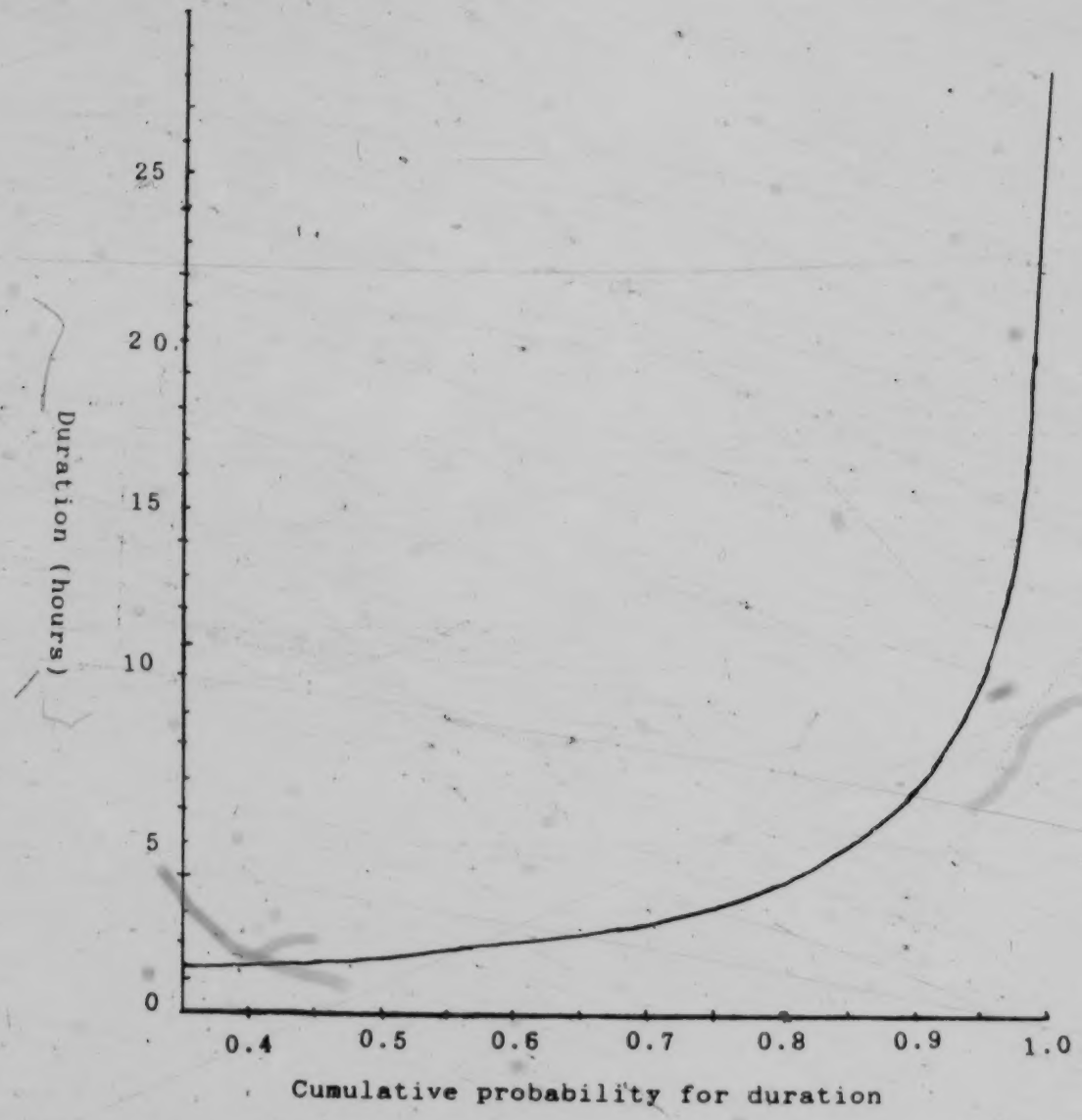


Figure 8 - Cumulative Probability Distribution of Duration

The duration is shown as a function of the cumulative probability of the occurrence of the duration. Note that the duration has an upper limit of 28 hours.

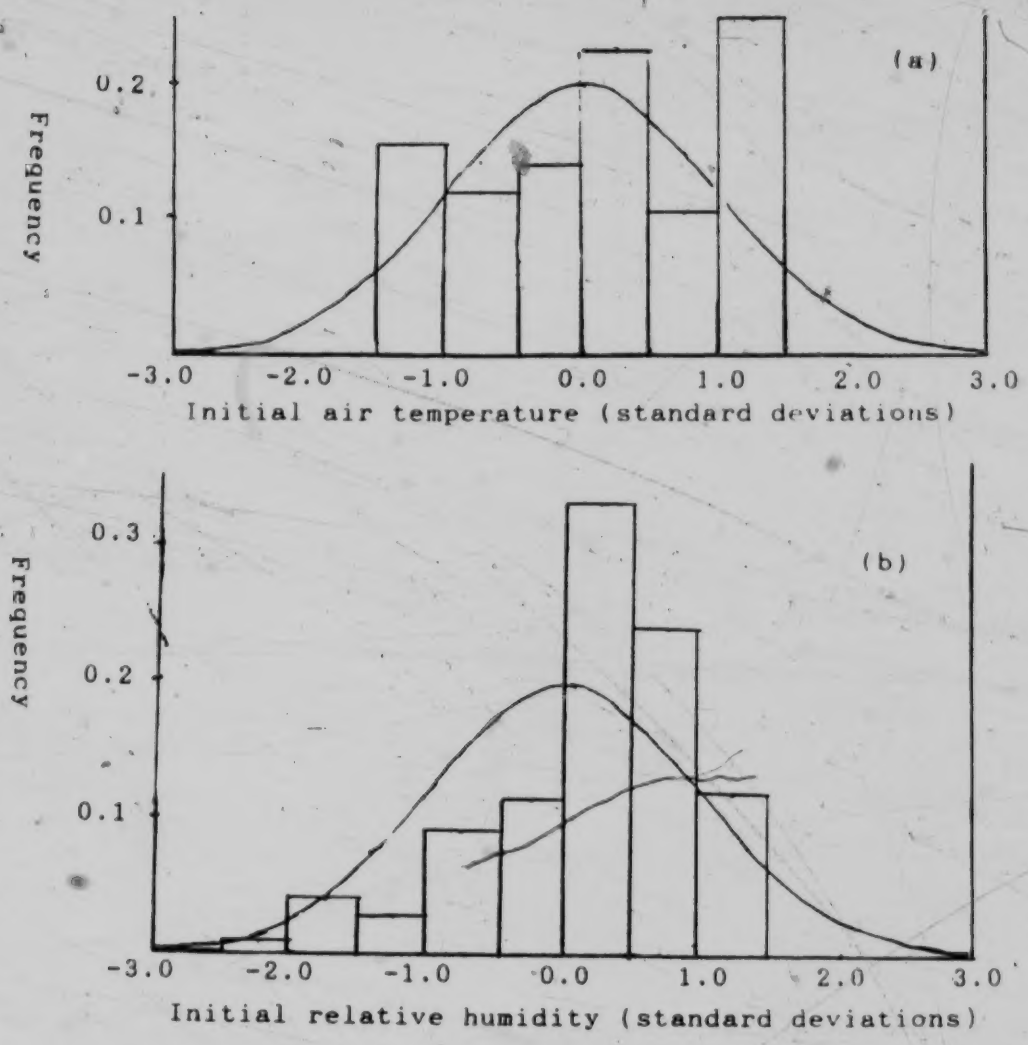


Figure 9 - Frequency Distributions of the Initial Values of Air Temperature and Relative Humidity

The values of the air temperature (a) and the relative humidity (b) are sorted according to their number of standard deviations from the mean. The initial air temperature has a mean of 2.2 ^{o}C, and a standard deviation is 2.5 ^{o}C. The initial relative humidity has a mean of 88.8%, and a standard deviation of 9.0%. A normal frequency distribution is shown for comparison.

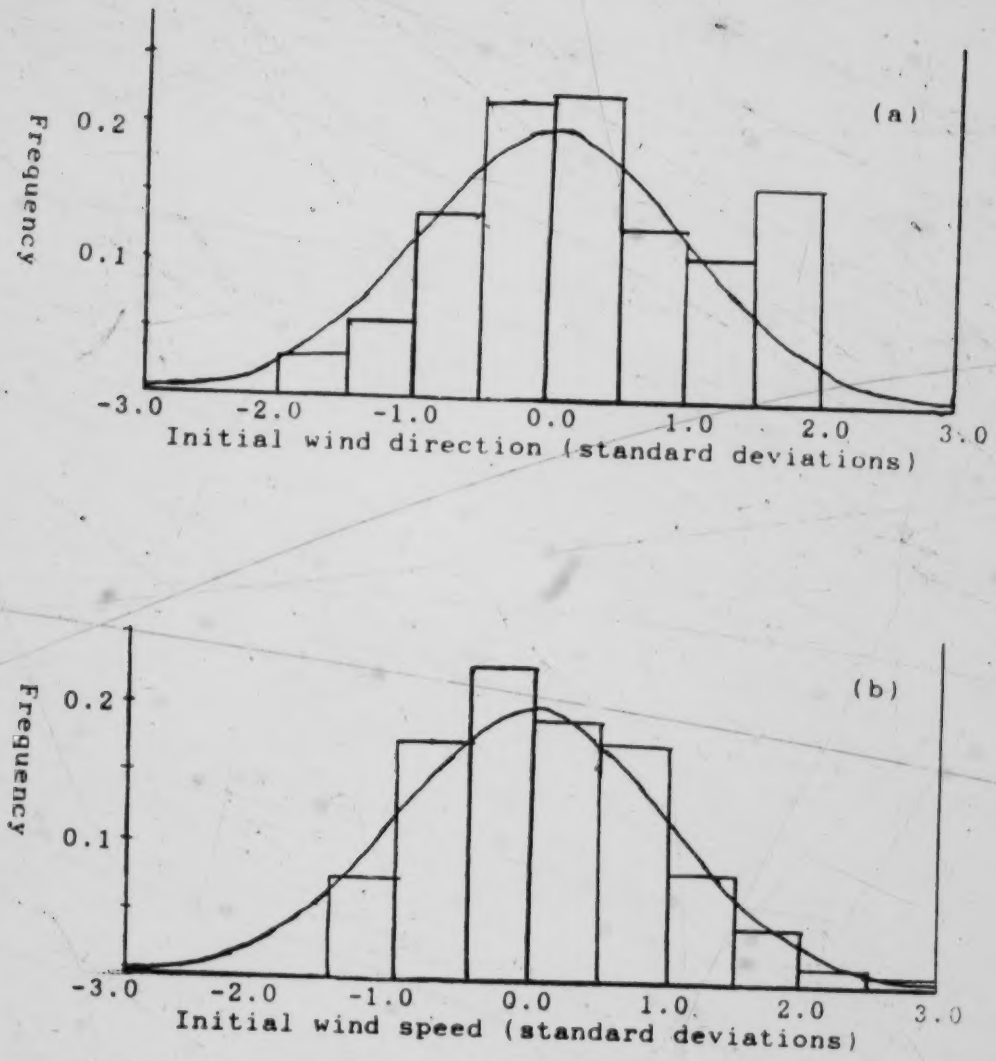


Figure 10 - Frequency Distributions of the Initial Values of Wind Direction and Wind Speed

The values of the wind direction (a) and the wind speed (b) are sorted according to their number of standard deviations from the mean. The initial wind direction has a mean of 356 degrees, and a standard deviation of 80 degrees. The initial wind speed has a mean of 4.9 m/s and a standard deviation of 3.0 m/s. A normal frequency distribution is shown for comparison.

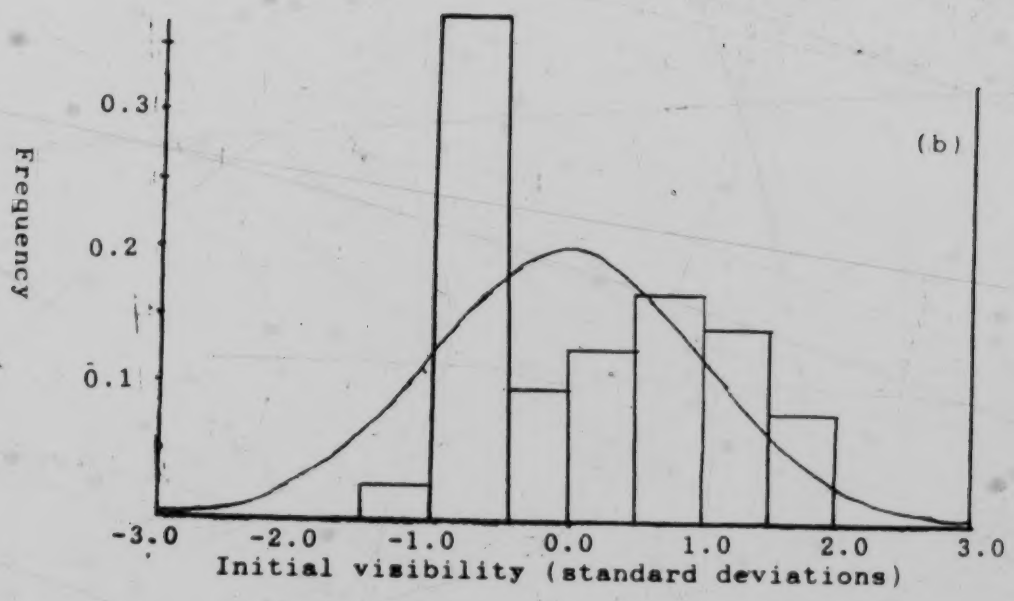


Figure 11 - Frequency Distributions of the Initial Values of Visibility

The initial values of the visibility are sorted according to their number of standard deviations from the mean. The initial visibility has a mean of 11.4 km, and a standard deviation of 8.8 km. A normal frequency distribution is shown for comparison.

the distributions were Gaussian. In the case of the relative humidity, with an upper bound of 100%, a Gaussian distribution, which is unbounded, is not even a qualitatively accurate representation of the natural distribution. However, Gaussian distributions were found to fit the initial conditions better than other simple distributions, so they have been used as first approximations. Section 4.1.4 examines the initial values of the meteorological parameters more thoroughly.

The hourly change in each parameter was examined next. A time increment of one hour was used because that was the smallest time step used in the AES data. The hourly changes were examined for trends (time dependency), means, standard deviations, and correlations with each other and with the value for the preceding hour. In all cases there was found to be a large degree of independence (see Section 4.1.5). Consequently the change in a parameter could be described by only its mean change and its standard deviation from the mean change. The distributions of hourly changes (see Figures 12 to 14) were also non-Gaussian. Too much of the data was grouped too closely to the mean for the distributions to be Gaussian. However, as a first approximation, the distributions were taken to be Gaussian. Section 4.1.5 examines the hourly changes more thoroughly.

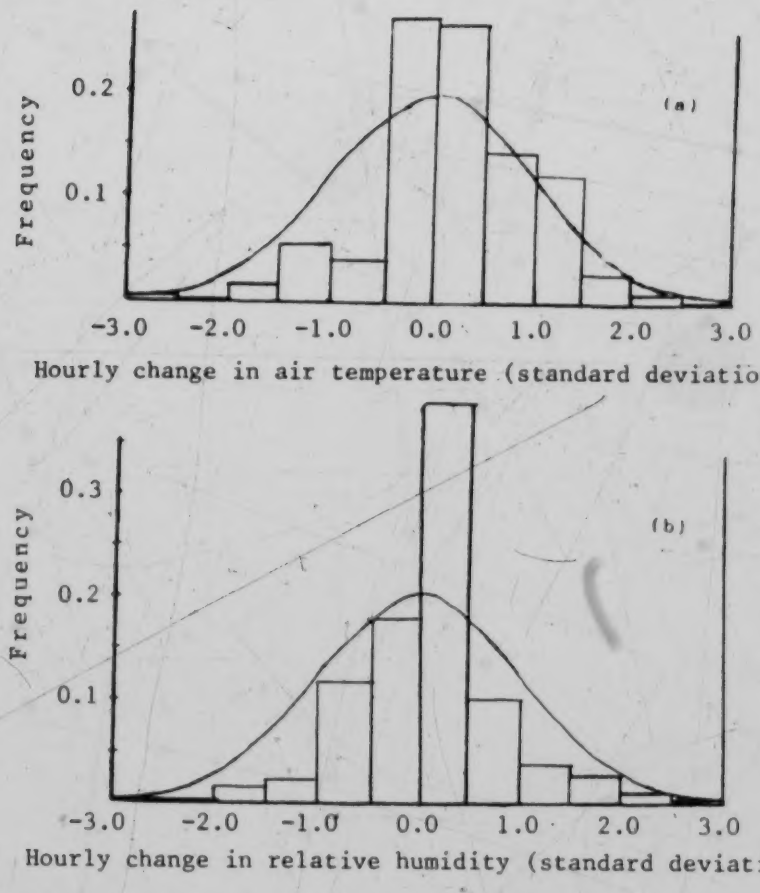


Figure 12 - Frequency Distributions of the Hourly Changes in Air Temperature and Relative Humidity

The hourly changes in the air temperature (a) and the relative humidity (b) are sorted according to their number of standard deviations from the mean. The hourly change in air temperature has a mean of 0.001, and a standard deviation of 0.656 ^{o}C. The hourly change in relative humidity has a mean of 0.443, and a standard deviation of 4.77%. A normal frequency distribution is shown for comparison.

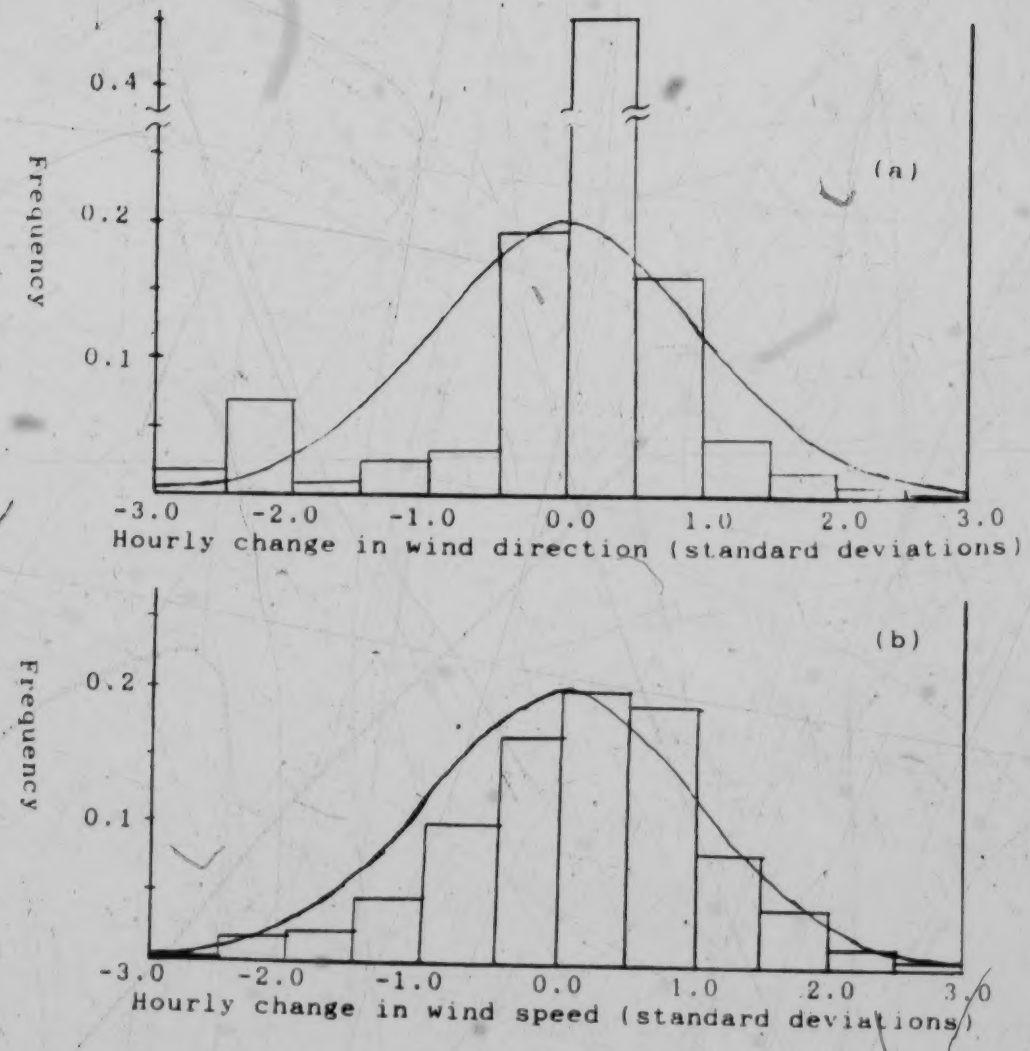


Figure 13 - Frequency Distributions of the Hourly Changes in Wind Direction and Wind Speed

The hourly change in the wind direction (a) and the wind speed (b) are sorted according to their number of standard deviations from the mean. The hourly change in wind direction has a mean of -3.349 degrees, and a standard deviation of 39 degrees. The hourly change in wind speed has a mean of 0.011 m/s and a standard deviation of 1.6 m/s. A normal frequency distribution is shown for comparison.

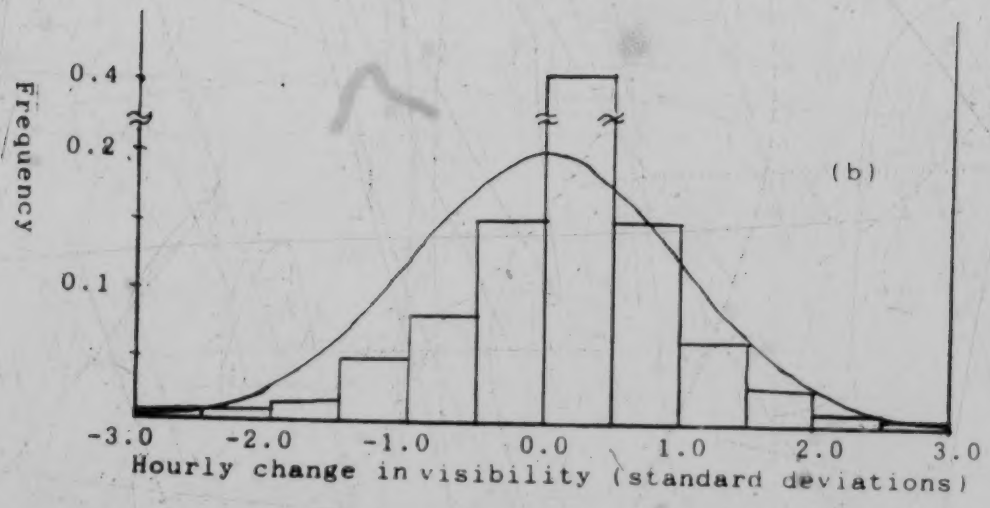


Figure 14 - Frequency Distributions of the Hourly Change in Visibility

The hourly changes in the visibility are sorted according to their number of standard deviations from the mean. The hourly change in visibility has a mean of -0.578 km, and a standard deviation of 6.39 km. A normal frequency distribution is shown for comparison.

4.1.1 Visibility as a Measure of Precipitation Rate

The hourly rate of precipitation for snow and for wet snow is not measured at AES surface stations. However, the rate of precipitation is an important parameter in accretion models. Consequently the rate of precipitation must be inferred from other measured data. Stallabrass and others have undertaken studies (Stallabrass, 1976) of the use of the visibility to estimate the precipitation rate during snowfall. Stallabrass determined the following relationship based on twelve years of snowfall and visibility data from the Toronto International Airport:

$$\log(V') = -0.419 - 0.607 \log(PR'), \qquad (37)$$

where V′ is the visibility in units of statute miles, and PR′ is the rate of precipitation in units of inches per hour.

This can be rearranged and converted to metric units:

$$PR = \text{Antilog}[\, 0.055 - \log(V) / 0.607 \,], \qquad (38)$$

where V is the visibility in units of kilometres, and PR is the rate of precipitation in units of centimetres per hour (which, for dry snow, is approximately equivalent to units of millimetres of water equivalent per hour).

Visibility is measured every hour at AES surface stations, so precipitation rate can be easily estimated from visibility (see Figure 15).

Wasserman and Monte (1972) have also proposed a method for estimating the precipitation rate from the visibility.

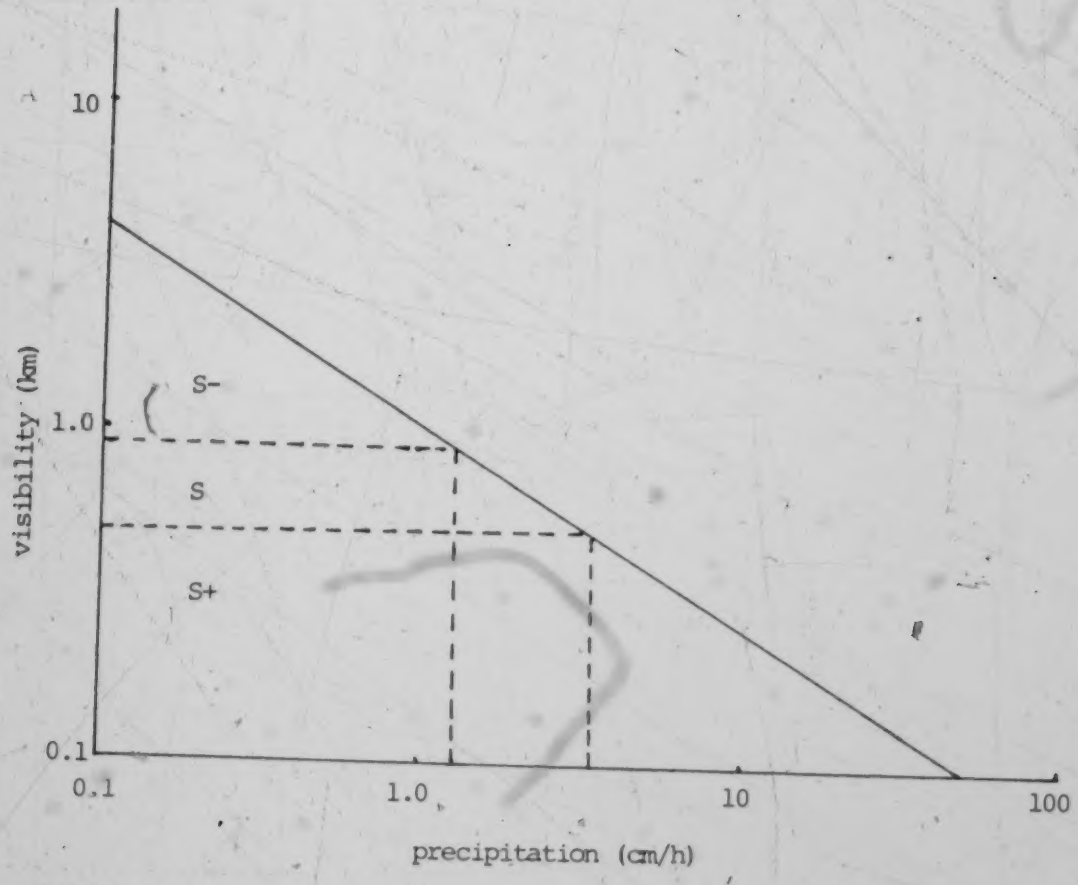


Figure 15 - Precipitation Rate as a Function of Visibility

The metric form of Stallabrass' linear relationship (Equation 38) between the log of the precipitation and the log of the visibility is shown. The areas representing light, moderate, and heavy snowfall are also shown. These make a good fit to Stallabrass' line.

This method is widely used. It is included in the weather manual used by the American Air Force (AWS pamphlet 105-56). Twenty years of precipitation data from LaGuardia Field in New York City were used to compare the snow intensity to the average hourly snowfall (Table 4).

Table 4 - Snow Intensity vs. Average Hourly Snowfall

Snow Intensity	Average Hourly Snowfall (cm/hr)	Probable Range of Snowfall Rate (cm/hr)
light (S-)	0.5	0 - 1.15
moderate (S)	2.5	1.2 - 3.5
heavy (S+)	4.0	> 3.5

The snow intensity can also be related to visibility (Table 5). Note that in 1972 visibility was measured in increments of an eighth of a statute mile.

Table 5 - The Relation of Snow Intensity to Visibility

Snow Intensity	Visibility
light (S-)	5/8 statute miles or more
moderate (S)	1/2 or 3/8 statute miles
heavy (S+)	1/4, 1/8, or 0 miles

Figure (15) shows that Equation (37) is completely compatible with Monte and Wasserman's method of determining the precipitation rate from the visibility. This implies

that Equation (37) is valid in different climatic regions. Consequently we take it to be valid in Alberta.

The data that Stallabrass used to determine a linear relationship between visibility and precipitation rate, had been previously used in a similar study (Stallabrass, 1976) by Richards. Richards did not assume a linear relationship between the visibility and the precipitation rate. Richards' curve is similar for light precipitation rates (less than 0.5 cm/hr). However, for heavy precipitation rates Stallabrass' linear relationship estimates precipitation rates double those of Richards. Stallabrass may have extrapolated Equation (37) into situations where it does not apply. Heavy precipitation comprised less than one percent of Richards data set (Stallabrass, 1976). Consequently, if this portion of the data was not heavily weighted in Stallabrass' least squared analysis, even a very good correlation would be insufficient to prove that Equation (37) is representive of the precipitation rate in low visibility situations. However, without Richards data set, or a similar data set, it is impossible to improve the accuracy of Equation (37). Consequently Equation (38), limited to visibilities greater than 100 m, is used in the extreme accretion model.

4.1.2 Estimation of Gust Speed

A very simple estimate of gust speed is used. The standard deviation of horizontal wind speed is approximated (for conditions of neutral stability in the surface layer)

by twice the friction velocity u_*. The conditions of neutral stability, an adiabatic lapse rate and no convection, are unlikely to be met during a wet snow event. However, the conditions should be close to that of neutral stability (Stull, 1988). The square of the quantity u_* is defined as the Reynolds stress divided by the average density of the air in the layer (Stull, 1988). The friction velocity is independent of height within the surface layer. The average wind speed as a function of height is assumed to be a log wind profile as described by K-theory:

$$U(z) = \frac{u_*}{k_v} \ln(z / z_o), \qquad (39)$$

where k_v - von Karman's constant: 0.4,

z - the height,

z_o - roughness length.

This can be rewritten to solve for u_*, with a known average speed at a reference height (z_{ref}):

$$u_* = \frac{k_v \, U(z_{ref})}{\ln(z_{ref}/z_o)}. \qquad (40)$$

If the maximum gust speed (v_g) is assumed to be 'n' standard deviations greater than the average speed, then this speed may be described by:

$$v_g(z) = \frac{U(z_{ref}) \ln(z)}{\ln(z_{ref})} + \frac{n \, 2 \, k_v \, U(z_{ref})}{\ln(z_{ref} / z_o)}. \qquad (41)$$

The reference height for AES measurements is ten metres. For prairie conditions, if the vegetation is not buried by snow, the roughness length can be approximated as three centimetres. Equation (41) can be solved for the gust

velocity in terms of the height of the line and the number of standard deviations:

$$v_g(z) = U(10\ m)\ (\ 0.434\ \ln(z) + 0.137\ n\ ). \quad (42)$$

A gust speed two standard deviations greater than the average wind speed was used in the model. If the frequency distribution of gust speeds was Gaussian then gusts of this speed or greater would occur for an average of ninety seconds each hour. The gust speed is not the maximum speed of a gust. It is a speed that will be maintained for several seconds. Both the wind speed and the gust speed increase with height, so the height of the highest transmission line on a tower (see Figure 16) is used to determine these speeds. The height of the highest transmission lines of a typical tower in southern Alberta is 31.6 m (100.5 ft).

So $$v_g(36.1\ m) = 1.761\ U(10\ m). \quad (43)$$

The assumptions involved in determining the above gust speed were applicable to prairie conditions, without a thick layer of snow. Under these conditions they make an excellent first approximation. However, these assumptions cannot be made for mountainous terrain. Neither the wind speed profile nor the roughness length would be appropriate. The log wind profile is applicable for terrain where an infinite plane can be approximated. For other terrain types a study of the site may be required.

When the snow completely covers the vegetation, the roughness length (z_o) will be reduced. Consequently, as the drag effects of vegetation decrease, the absolute value of

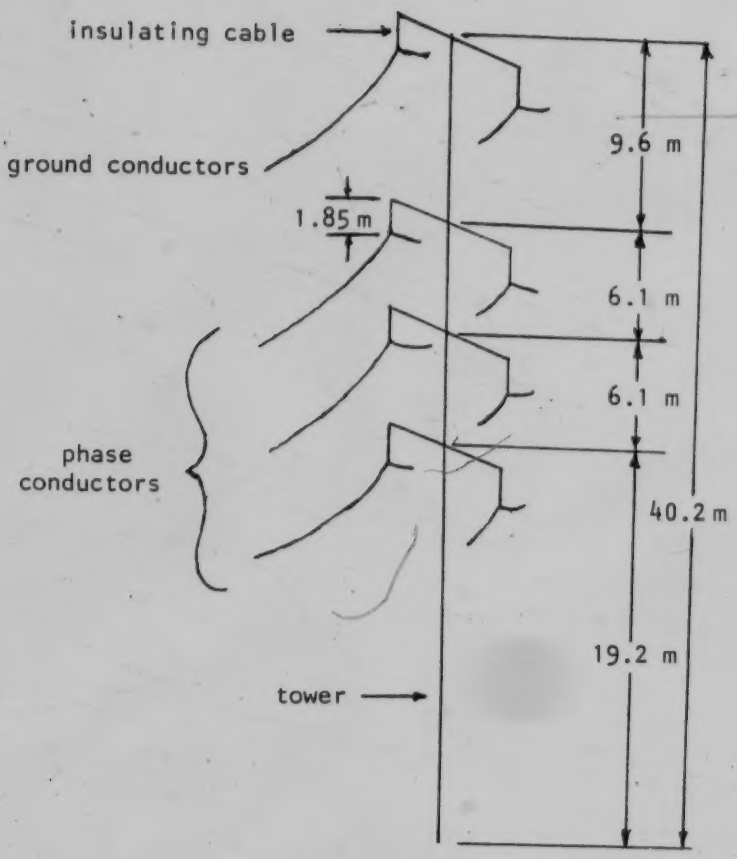


Figure 16 - Transmission Line Tower

The placement and heights of the transmission lines on a L-tower, typical in rural, southern Alberta, are shown. The electrical current is carried by the three sets of phase wires.

the log of the roughness length is increased, and the speed of the gust is decreased. However, the reduced frictional drag results in increased wind speeds (Stull, 1988). The effect of snow cover on the roughness length can have a large effect on the wind and gust speeds.

4.1.3 Initial values of the meteorological parameters

The initial values of the meteorological parameters are effectively independent of each other. The correlations between the initial values of the meteorological parameters are poor, so each parameter can be treated as independent of the others. Table 6 lists these correlation coefficients. The data from all 464 potential wet snow events were used to determine these correlation coefficients, although only the 201 precipitation events with snow were used for visibility. Correlation coefficients greater than approximately 0.16 indicate that there is probably a correlation (Taylor, 1982). However, the fraction of the variance explained by the correlation is equal to the correlation coefficient squared (Taylor, 1982). Consequently these correlations are of little use unless the correlation coefficients are greater than 0.64 (explaining approximately 40% of the variance).

Table 6 - Correlation Coefficients Between Initial Values of the Meteorological Parameters

	Ta	RH	V	U	Dir
Air Temperature	1.000	---	---	---	---
Relative Humidity	-0.084	1.000	---	---	---
Visibility	0.187	-0.535	1.000	---	---
Wind Speed	0.008	-0.079	-0.083	1.000	---
Wind Direction	-0.079	-0.073	-0.003	0.345	1.000

Most of these correlations coefficients have a magnitude less than 0.1. This means that the parameters are largely independent of each other. The only correlation with a magnitude greater than 0.5 is that between visibility and relative humidity. However, the correlation coefficient of -0.535 is low enough in magnitude (less than 0.64) to ignore. Therefore, in the model of the meteorological parameters, all the initial conditions of the parameters were treated as independent.

The observed distributions for all of these parameters are non-Gaussian (Figures 10 to 12). The distribution for wind direction was found by altering the break point (initially $0^{o}/360^{o}$) in increments of ten degrees, and assuming that the best distribution had the lowest standard deviation. All the distributions are too tight about the mean to be Gaussian. A reduced chi squared test (Taylor, 1982) was used to test the 'closeness of fit' to a Gaussian distribution. The data were split into twelve bins that

should have held equal numbers of points if the distributions were Gaussian. Reduced chi squared values (Table 7) were produced by comparing the number of data values in each bin to the number expected if the distribution were Gaussian. The chi squared value should be approximately one or less for a good fit. In the case of ten degrees of freedom (twelve bins and two constraints) a reduced chi squared value of unity would mean that, if the distribution actually is Gaussian, there is a 44% chance that it could have reduced chi squared value greater than unity.

Table 7 - Distribution Statistics of Initial Conditions

Parameter	Mean	Standard Deviation	Reduced Chi Squared
Air Temperature [oC]	2.2	2.5	15.7
Relative Humidity [%]	88.8	9.0	12.3
Visibility [km]	11.4	8.8	15.6
Wind Speed [m/s]	4.9	3.0	4.9
Wind Direction [degs]	356	80	8.9

None of the distributions are extremely well approximated by a Gaussian distribution. Given the large reduced chi squared values, it is unlikely that the distributions are Gaussian. Nevertheless, in the wet snow accretion model, all these distributions are approximated as Gaussian, and hence they can be described with a mean and a standard deviation (Table 7). The approximation that the

distributions of initial meteorological parameters are Gaussian distributions is improved by limiting the ranges of the meteorological parameters. The relative humidity is limited at both extremes. The relative humidity cannot be less than zero percent, and cannot be greater than one-hundred percent. Similarly the visibility cannot be less than zero kilometres or greater than twenty-five kilometres (AES treats 25 km as the upper limit, equivalent to the horizon). In the extreme accretion model, the lower limit of visibility is one tenth of a kilometre. This lower limit is necessary because of the relationship between the visibility and the precipitation rate. The precipitation rates corresponding to visibilities less than one tenth of a kilometre are unreasonably large (50 mm water equivalent per hour). Further, the wind speed must be constrained to be equal to or greater than zero. With the exception of the upper limit on visibility, similar restrictions would apply in nature.

The recorded direction of the wind is relative to a wind flowing from the North. Thus a direction of 0^{o} is northerly, and a direction of 90^{o} is easterly (from the East). The extreme wet snow accretion model requires the initial direction of the wind relative to the axis of the transmission line. Obviously, the mean initial wind direction and the alignment of the transmission lines will vary throughout Alberta. It seems unlikely that the companies building transmission lines would want to have a

set of standards for the failure strengths of transmission lines that would vary with the location and the alignment of the line. Therefore the worst case (resulting in the largest accretions) is examined. This occurs when the wind direction is perpendicular to the axis of the transmission line. It seems unreasonable to model the initial wind direction as always being at right angles to the line. Therefore, in the accretion model the mean initial direction is set at ninety degrees.

4.1.4 Persistence

Examination of the hourly data from CFB Namao showed there was an unusually large number of hourly changes in meteorological parameters that were equal to zero. Even considering that the distributions are non-Gaussian, with a greater than normal number of small changes, for each parameter the number of hourly changes equal to zero is much larger than expected. The expected percentage of hourly changes equal to zero was determined based on the assumption that the distributions are Gaussian, and that the measurements were rounded to the nearest increment (i.e. if temperature is recorded in tenths of a degree, then measurements were rounded to the nearest tenth of a degree). Consequently the fraction of changes rounded to zero is approximately equal to the fraction of changes that are less than one half of a scale increment. Table 8 lists both the observed percentages and the expected percentages of hourly changes that were equal to zero. Some of the difference,

between the observed percentages and the expected percentages, is probably caused by 'observer inertia'. If there was a small change, the observer might have simply repeated the previous observation, even if the change was large enough to change the value. The differences between the observed and the expected percentages (assuming a normal distribution) are great enough that persistence is considered in the model of the meteorological parameters.

Table 8 - Mean Likelihood of Persistence

	Observed	Expected	Scale Increments
Temperature [oC]:	34%	7.2%	0.1
Relative humidity [%]:	24%	8.0%	1.0
Visibility [km]:	25%	0.72%	0.1
Wind speed [m/s]:	18%	0.003%	0.01
Wind direction [deg]:	39%	12.9%	10.0

The fraction of hourly changes that are equal to zero depends on the sensitivity with which the phenomenon is measured and recorded. Temperature, relative humidity, visibility, and wind direction are all recorded to a similar accuracy. They are recorded in increments equal to between 3% and 0.4% of their typical range. Wind speed is recorded in increments of approximately 0.01% its typical range. This is a partial explanation of why wind speed has a smaller fraction of hourly changes that are equal to zero.

The percentage of the hourly changes in the meteorological parameters that were equal to zero did not vary significantly with the time from the onset of the precipitation event. Table 9 lists the correlations and the functions of the best fit curves, for each meteorological parameter, for the percentage of hourly changes equal to zero as a function of time.

Table 9 - Time Dependence of Hourly Changes Equal To Zero

Meteorological Parameter	Correlation Coefficient	Function (t in hours)
Air temperature [^{o}C]	0.485	$\delta_{Ta} = \exp(5.88\ t + 28.04)$
Relative humidity [%]	0.511	$\delta_{RH} = \exp[0.16\ \ln(t) + 2.99]$
Visibility [km]	0.423	$\delta_{Vis} = \exp(203\ t^2 + 443)$
Wind speed [m/s]	0.263	$\delta_{U} = \exp(78.8\ t^2 + 218.5)$
Wind direction [deg]	0.781	$\delta_{Dir} = \exp(0.217\ t^{0.5} + 3.202)$

The only relationship with a correlation coefficient great enough to be significant is the wind direction relationship. Table 10 shows observed percentages of the changes in wind direction that are equal to zero, and compares them with percentages predicted by the relationship in Table 9.

Table 10 - Hourly changes in Wind Direction Equal to Zero

	Time (hours)											
	1	2	3	4	5	6	7	8	9	10	11	12
Predicted	30		36	38	40	42	44	45	57	49	50	52
Observed	34		31	35	51	42	41	46	39	60	54	47

Table 10 shows that the equation for time dependence of the percentage of hourly changes that are equal to zero for wind direction in Table 9 is not a good predictor. In the meteorological parameter model the percentage of hourly changes equal to zero is treated as independent of time for all parameters.

A small effort was made to determine whether or not hourly changes of meteorological parameters equal to zero were more likely to occur under certain conditions. There are two situations where this seems most likely to be the case. The first occurs when the air temperature at the beginning of the hour equals the freezing point of water, zero degrees Celsius. The second occurs when, over one hour, the temperature remained constant. Both conditions imply that the air mass might be more stable (see Table 11).

Table 11 - Fraction of Hourly Changes Equal to Zero

Special Conditions

Hourly change	none	Hourly Change in $T_a = 0$	$T_a = 0^oC$
Air Temperature	34%	100%	55%
Relative Humidity	24%	37%	32%
Visibility	25%	30%	21%
Wind speed	18%	20%	18%
Wind direction	39%	44%	50%

The condition that the hourly change in temperature be equal to zero (Table 11) has little effect on the percentage of hourly changes that are equal to zero. Only the percentage for relative humidity is raised significantly. However, the percentages for all parameters are increased. The large increase in the chance of persistence in relative humidity occurs because of the connection between the relative humidity and the air temperature. The relative humidity is the ratio of the vapour pressure to the saturation vapour pressure. If the relative humidity changes, then either the vapour pressure or the saturation vapor pressure must have changed. The saturation vapour pressure is a function of the air temperature. Consequently when there is no change in the air temperature, there is no change in the saturation vapour pressure. Thus, when the temperature is constant, the relative humidity can change only if the vapour pressure changes. The condition that there is no change in temperature over one hour eliminates

one of the mechanisms for a change in relative humidity. The correlation (see Section 4.1.5) between the hourly changes in relative humidity and the hourly changes in visibility may explain the increase in percentage of changes in visibility that are equal to zero. In the extreme accretion model these effects have been ignored.

The condition that the air temperature at the end of the hour be equal to $0^{o}C$ (Table 11) has an apparently significant effect on three of the parameters. The effect on the percentage of hourly changes that are equal to zero for visibility and wind speed is negligible (5% or less). The effect on air temperature is the greatest: an increase of twenty-one percent. This is not unexpected because an isothermal layer with a temperature of zero degrees Celsius can be expected to form in or near a melting layer (Stewart, 1984; Stewart and King, 1987). The air in these layers is saturated, and consequently falling snow does not exchange heat with the air through sublimation. Similarly, because the ambient air temperature is at zero degrees, there is no melting or freezing occurring on the snowflakes. Heat exchange due to convection is ignored because to snowflake is assumed to move with the wind. Since no heat is being exchanged between the falling snow and the ambient air, there is less chance of a change in temperature occurring. The effect on the percentages for relative humidity and for wind direction is smaller (8% and 11% respectively), but they may be significant.

The unconditional percentage of hourly changes that are equal to zero can be represented statistically as a Kronecker delta function. This approach is used in the extreme value model. If a randomly generated number (generated from a uniform distribution between zero and one) is less than or equal to the fraction of changes that are equal to zero, then the hourly change is set equal to zero. The fractions of hourly changes equal to zero were determined before any changes due to time-dependent trends were considered. The hourly changes equal to zero were removed from the data set for the remaining calculations. Consequently any hourly change of a meteorological parameter can be represented as a delta function times the other changes. Neither of the special conditions affecting the delta function are considered in the model.

4.1.5 Time-Dependent Trends

Several of the meteorological parameters were found to be time-dependent. A curve fitting routine (Appendix G) was used to run a least squares analysis between time (in hours) and various functions (square, square root, log, natural log) of the mean hourly changes, sorted according to the number of hours from the onset of the precipitation event. Only data from the first twelve hours of each event were used. There were too few data points, for later hours, to establish a good mean hourly change. A detailed listing of the results is given in Appendix E. Table 12 list the

correlation coefficients and formulas for the trends. Plots of these trends are given in Appendix E.

Table 12 - Time-Dependent Trends

Parameter	Correlation Coefficient	Trend Formula (t in hours)
Temperature [^{o}C]	-0.869	$\Delta T_{at} = -[0.156 - 0.066 \ln(t)]$
Relative humidity [%]	-0.914	$\Delta RH_t = 3.172 - 1.955 \ln(t)$
Visibility [m]	-0.563	$\Delta V_t = [2.299 - 0.787 \ln(t)]^{0.5}$
Wind speed [m/s]	0.485	$\Delta U_t = (0.008\ t + 0.005)^{0.5}$
Wind direction [deg]	0.217	$\Delta dir_t = [28.52 \ln(t) + 18.54]^{0.5}$

Several of these trends were ignored in the accretion model. The trend for wind direction was ignored because of the small correlation coefficient (magnitude less than 0.64). The trend for wind speed also has a low correlation coefficient, but it can be ignored for other reasons as well. The trend in wind speed is insignificantly small compared to the magnitude of the wind speed and its standard deviation. Similarly the trend for visibility is also small, especially early in the event. Table 13 compares the time-dependent trends of hourly changes, of the first, second, third, fifth, and tenth hours of an event, to the standard deviations of the hourly changes (for all hours) of the observed parameters. Most of the mean hourly changes in meteorological parameters, due to the time-dependent trends, are small compared the standard deviations of the hourly changes. Consequently the hourly changes due to the trends are small compared to most hourly changes. Only the time-

dependent trends for the hourly changes in air temperature and relative humidity were used in the accretion model, as they are the only ones that have trends greater than one quarter of their corresponding standard deviations. The time-dependent trends of hourly changes for the other parameters are approximated as zero for all hours.

Table 13 - Time-Dependent Trends

	Standard	Hourly changes				
Parameters	Deviation	1st	2nd	3rd	5th	10th
Temperature [°C]	0.671	-0.395	-0.332	-0.289	-0.223	-0.063
RH [%]	4.992	3.172	1.817	1.024	0.026	0.000
Visibility [km]	6.391	1.516	1.324	1.198	1.016	0.698
Wind Speed [m/s]	1.589	0.145	0.170	0.192	0.230	0.305

The magnitude of the function for the trend in the hourly changes for wind speed increases with time. The increase is not bounded; the trend does not approach a limit. This is because the data for the correlations used in determining these formulas for the trends used only the mean hourly changes for the first twelve hours of potential wet snow events. It is unreasonable to assume that the formulas are a good representation of the trends at the later stages of long precipitation events. Both the trends for changes in air temperature and changes in relative humidity are monotonic, and in the early hours of the potential wet snow events they approach zero. When the trends reach zero (eleven hours for air temperature, and six hours for

relative humidity), the time-dependent hourly changes are treated as zero for the rest of the event. With this approximation the equations for the time-dependent trends are used only at times when they are appropriate.

4.1.6 Statistics of Hourly Changes

Meteorological parameters such as temperature and relative humidity are recorded in time steps of one hour at AES surface stations. Airports report special updates at anytime when the changes in meteorological conditions warrant the report. However, these reports are not on AES' list of magnetic tapes. This means that the minimum convenient time step for these parameters is one hour. Therefore the hourly changes in these parameters have been examined. The recorded values are the values of the parameters at the time when the data was recorded; they are not hourly averages. As with the initial conditions, the correlations (Table 14) between the hourly changes of the various parameters are poor and consequently can be ignored. The correlation coefficients (Table 14) between the parameter at the beginning of the hour and the change in the same parameter are also low.

Table 14 - Correlations Coefficients of Hourly Changes

Hourly Changes	Hourly Changes					Raw
	ΔT_a	ΔRH	ΔV	ΔU	ΔDir	Parameter
ΔT_a	1.000	---	---	---	---	-0.254
ΔRH	-0.084	1.000	---	---	---	-0.451
ΔV	0.141	-0.259	1.000	---	---	-0.081
ΔU	0.021	-0.005	-0.025	1.000	---	-0.185
ΔDir	-0.046	-0.032	-0.085	0.043	1.000	0.105

The temporal trends were subtracted from the values of air temperature and relative humidity before the correlation coefficients were evaluated. No trends were subtracted from the other variables because the temporal trends for the other variable have been assumed to be negligible.

The distributions for hourly changes (Figures 12 to 14) were similar to those for the initial parameters. They were bell curved but were not Gaussian. These distributions also had a super-Gaussian peak. A reduced chi squared test was applied in the same manner as with the distributions of initial values. In each case the reduced chi squared value (Table 15) was of the order of one hundred. This means that it is highly unlikely that the distributions are Gaussian. Nevertheless for convenience, these distributions were approximated as Gaussian. With this approximation, the probability distributions (which are by definition normalized) can be described with only their means and standard deviations (Table 15).

Table 15 - Distribution Statistics of Hourly Changes

Parameter	Mean	Standard Deviation	Reduced Chi Squared
Air Temperature [°C]	0.001	0.656	97.858
Relative Humidity [%]	0.443	4.774	112.420
Visibility [km]	-0.578	6.391	103.432
Wind Speed [m/s]	0.011	1.598	124.419
Wind Direction [deg]	-3.349	39.496	99.449

In the extreme accretion model, the meteorological parameters must have the same ranges that they possess in nature. A change in a parameter cannot be large enough to alter the value of the parameter so that it is outside the natural range of the parameter. These ranges are the same ranges discussed at the end of Section 4.1.3. The imposition of these range restrictions forces the probability distributions to be more accurate.

The change in a parameter could be determined by using the parameter's Gaussian probability distribution. The probability distribution can be written as a cumulative probability distribution. The cumulative probability has a minimum of zero, and a maximum of unity. A random number generator, with a uniform distribution between zero and one, is used to generate a cumulative probability. The change in the parameter is the change corresponding to that cumulative probability. An equivalent technique, that is easier to program, is to use a normal probability distribution. For a

normal distribution the value corresponding to the cumulative probability is a certain number of standard deviation from the mean. This number of standard deviations is multiplied by the standard deviation of the hourly changes of a parameter, to determine the value of the hourly change for that parameter. If the change in the parameter would cause the value of the parameter to lie outside the accepted range of the parameter, then one of two things would happen. Usually the change is ignored, and new changes are generated until an acceptable value is found. An exception to this occurs if the relative humidity or the visibility increase beyond their upper limits. In this case, the parameter is set equal to its upper limit. In the case of visibility, this is an accurate simulation of the recording technique (i.e. if the visibility is greater than twenty-five kilometres then it is recorded as twenty-five kilometers). In the case of relative humidity, this is an accurate simulation of saturation; in other words, the relative humidity increase until it reaches 100%, and then it can increase no further. The limitations on the ranges of the parameters makes the accretion model's simulation of these input meteorological parameters much more realistic.

4.2 Results of a risk analysis on extreme line loads

A risk analysis was performed on the extreme annual values produced by a modified version of Finstad's wet snow accretion model (Finstad, 1989). The annual maxima of accretion mass, vertical load (accretion weight), horizontal load due to the hourly average wind speed, and the horizontal load due to gusts, were simulated for the equivalent of one hundred years, on a line 20.38 mm in diameter, 370 m long, with a torsional rigidity of 0.1 N/m. This is typical of the transmission lines in southern Alberta. The sorted extremes are listed in Appendix D. The analyses for accretion mass and vertical load are similar because they are related by the gravitational acceleration. The horizontal loads are more difficult to relate because the model treats the gust velocity as equal to the average velocity plus a constant. Despite their similarities all four extremes were examined.

A linear regression was performed on the extremes and their reduced variates. The extreme average wind load and extreme gust load had a high correlation (0.893) with the reduced variate. The exponential distribution is a good fit for the generated extremes of average wind and gusts (Table 16). However, it is not a good fit for the mass or the vertical load (Table 16). Nevertheless, for lack of a better distribution, the exponential distribution is also used for these extremes.

Table 16 - Extreme Value Statistics

	Mass (kg/m)	Vertical Load (N/m)	Wind Load (N/m)	Gust Load (N/m)
Slope: a	1.259	12.354	3.028	4.181
Standard Deviation of a	.185	1.815	0.168	0.232
Y-Intercept: u	-.419	-4.109	2.475	3.417
Standard Deviation of u	.255	2.502	0.231	0.319
Correlation Coefficient	0.599	0.599	0.893	0.893
Number of Data Points	85	85	85	85

The constants 'u' and 'a' describing the distributions of extremes have been calculated using two methods: linear regression and the method of moments. The method of moments determines 'u' (Equation 21) through the similarity to a mean for the distribution of extremes. The 'u' determined by regression is the y-intercept in the linear relationship between the extremes and the reduced variates. The method of moments examines 'a' as a measure of the dispersion of the extremes, and uses equation (20) to determine 'a'. However, 'a' is also the slope of the linear regression relationship between the extremes and the reduced variates. It should be noted that in Table 17 the constants for vertical load differ from those for mass by a factor equal to gravitational acceleration (9.81 m/s^2). This multiplicative factor carry through to determining estimates of extremes as a function of mean return time (Table 18), as can be seen

from Equation (22). This information may be useful when a quick estimation of extremes is needed.

Table 17 - Constants 'a' and 'u'

	Regression Estimates		Method of Moments	
	a	u	a	u
Mass	1.2593	-0.4180	1.6728	-0.4312
Vertical Load	12.3540	-4.1087	16.4099	-4.2306
Wind Load	3.0281	2.4745	2.7485	2.6579
Gust Load	4.1807	3.4166	3.7943	3.6698

Given the parameters 'a' and 'u', it is possible to determine the extremes corresponding to any mean return period or any probability of occurrence in a particular time period. This is a study of annual extremes, so the time periods are in units of years. Table 18 shows the extremes for masses and for vertical loads. The extremes determined from the parameters derived by both the regression method and the method of moments are listed. Kinnison (1985) suggests that the method of moments may be the more accurate of the two techniques. This method produces the larger extreme values of vertical load for any mean return period of three years or greater. Because of the poor linear correlation between the extremes and the reduced variates, there is a large difference between the constants, 'u' and 'a', produced by the different methods. Hence there is a large difference between the extreme values determined by

each set of constants. With the data available it is not possible to determine which set of extremes is a better representation of nature. However, there are three reasons to use the set derived from the method of moments. The first is Kinnison's suggestion that the method is more accurate. The second is the poor correlation of the linear regression. The third is that this set represents the worst case, and it is better to err on the side of caution.

Table 18 - Extreme Values as a Function of the Mean Return Period

Return Period (yrs)	Prob of Exceed.	Reduced Variate	Accretion Mass Extreme Values Reg. (kg/m)	Accretion Mass Extreme Values Mom. (kg/m)	Vertical Load Extreme Values Reg. (N/m)	Vertical Load Extreme Values Mom. (N/m)
2	.500	.367	.043	.182	.419	1.784
3	.333	.903	.718	1.079	7.043	10.583
4	.250	1.246	1.150	1.653	11.283	16.214
5	.200	1.500	1.470	2.078	14.422	20.383
6	.167	1.702	1.725	2.416	16.918	23.699
7	.143	1.870	1.936	2.697	18.991	26.453
8	.125	2.013	2.117	2.937	20.765	28.809
9	.111	2.139	2.275	3.147	22.315	30.869
10	.100	2.250	2.415	3.333	23.692	32.698
15	.067	2.674	2.948	4.041	28.923	39.645
20	.050	2.970	3.322	4.537	32.585	44.510
25	.040	3.199	3.609	4.919	35.406	48.257
30	.033	3.384	3.843	5.230	37.701	51.305
35	.029	3.541	4.040	5.492	39.635	53.875
40	.025	3.676	4.211	5.718	41.308	56.096
45	.022	3.795	4.361	5.918	42.780	58.052
50	.020	3.902	4.495	6.096	44.096	59.800
60	.017	4.086	4.727	6.404	46.369	62.819
70	.014	4.241	4.922	6.663	48.288	65.369
80	.013	4.376	5.092	6.888	49.949	67.575
90	.011	4.494	5.241	7.087	51.413	69.519
100	.010	4.600	5.374	7.264	52.721	71.257

Table 19 - Extreme Values as a Function of the Mean Return Period

Return Period (yrs)	Prob of Exceed.	Reduced Variate	Wind Load Extreme Values Reg. (N/m)	Wind Load Extreme Values Mom. (N/m)	Gust Load Extreme Values Reg. (N/m)	Gust Load Extreme Values Mom. (N/m)
2	.500	.367	2.712	2.694	5.621	5.583
3	.333	.903	3.712	3.659	7.693	7.583
4	.250	1.246	4.351	4.276	9.019	8.863
5	.200	1.500	4.825	4.733	10.001	9.811
6	.167	1.702	5.202	5.097	10.781	10.564
7	.143	1.870	5.514	5.399	11.430	11.190
8	.125	2.013	5.782	5.657	11.984	11.726
9	.111	2.139	6.016	5.883	12.469	12.194
10	.100	2.250	6.224	6.084	12.900	12.609
15	.067	2.674	7.013	6.846	14.536	14.189
20	.050	2.970	7.566	7.379	15.681	15.294
25	.040	3.199	7.991	7.790	16.563	16.146
30	.033	3.384	8.338	8.124	17.281	16.839
35	.029	3.541	8.630	8.406	17.886	17.423
40	.025	3.676	8.882	8.650	18.409	17.928
45	.022	3.795	9.104	8.864	18.870	18.372
50	.020	3.902	9.303	9.056	19.281	18.769
60	.017	4.086	9.646	9.387	19.992	19.456
70	.014	4.241	9.935	9.666	20.592	20.035
80	.013	4.376	10.186	9.908	21.112	20.537
90	.011	4.494	10.407	10.121	21.570	20.978
100	.010	4.600	10.604	10.312	21.979	21.374

The extremes for the average hourly horizontal wind load and the gust load are given in Table 19. In this case the extremes determined through the regression technique are more severe for all the listed mean return periods, and also for any greater mean return periods. The set, derived by the method of moments, may be more accurate (Kinnison, 1985), but it does not represent the worst (most severe) case. Consequently 'a' and 'u', derived by the regression method, are used in later analyses of the horizontal load.

The extreme values can be represented as a function of the risk and the designed lifetime of the transmission line. This is probably the most useful representation of the data in tabular form. In this form the minimum tolerance for which the transmission lines must be constructed, for a given risk and lifetime, is readily apparent. This representation is similar to that of Table 2, except that the minimum tolerance is expressed in terms of force on the line, rather than the mean return period of that force. Table 20 gives the minimum mass (kg/m) tolerances. It is probably more convenient to work in terms of force rather than mass. The wind drag and the force due to the movement of the line are also easily expressed in terms of force. Table 21 gives the minimum vertical load tolerances. It should be noted that these tables do not include the mass (0.7872 kg/m) or weight (7.722 N/m) of the transmission line. The weight of the line is not a large factor, but in the case of a moderate risk, moderate lifetime situation it

will be a significant consideration in designing transmission towers.

Tables 22 and 23 show the minimum horizontal tolerances for the average hourly wind load and the gust load as a function of the risk the and designed lifetime. The tolerance of transmission lines must be designed with respect to the gust load. The gust load is by definition greater than the average wind load. The table for the average wind loads exists only for the purpose of comparison with the gust loads. Both tables represent only the static situation. The load on the system can be greatly increased by the dynamics of the line (Lozowski and Gates, 1987). Because of this the gust table is, for practical purposes, inadequate for designing line and tower tolerances. It is, however, a start in the right direction.

Table 20 - Sufficient Design Mass Tolerance (kg/m)

Risk of Exceedance (%)	Designed Lifetime (years)							
	2	5	10	15	20	25	50	100
.75	.83	2.61	3.84	4.54	5.03	5.41	6.58	7.75
.50	1.65	3.33	4.54	5.23	5.72	6.10	7.26	8.43
.40	2.08	3.72	4.92	5.61	6.10	6.47	7.64	8.80
.30	2.61	4.22	5.41	6.10	6.58	6.96	8.12	9.28
.25	2.94	4.54	5.72	6.40	6.89	7.26	8.43	9.59
.20	3.33	4.92	6.10	6.78	7.26	7.64	8.80	9.96
.15	3.84	5.41	6.58	7.26	7.75	8.12	9.28	10.44
.10	4.54	6.10	7.26	7.94	8.43	8.80	9.96	11.12
.05	5.72	7.26	8.43	9.11	9.59	9.96	11.12	12.28
.02	7.26	8.80	9.96	10.64	11.12	11.50	12.66	13.82
.01	8.43	9.96	11.12	11.80	12.28	12.66	13.82	14.98

Table 21 - Sufficient Design Vertical Load Tolerance (N/m)

Risk of Exceedance (%)	Designed Lifetime (years)							
	2	5	10	15	20	25	50	100
.75	8.16	25.59	37.64	44.51	49.34	53.06	64.56	76.00
.50	16.21	32.70	44.51	51.31	56.10	59.80	71.26	82.67
.40	20.38	36.54	48.26	55.02	59.80	63.49	74.94	86.34
.30	25.59	41.43	53.06	59.80	64.56	68.25	79.67	91.07
.25	28.81	44.51	56.10	62.82	67.57	71.26	82.67	94.07
.20	32.70	48.26	59.80	66.51	71.26	74.94	86.34	97.73
.15	37.64	53.06	64.56	71.26	76.00	79.67	91.07	102.46
.10	44.51	59.80	71.26	77.94	82.67	86.34	97.73	109.12
.05	56.10	71.26	82.67	89.34	94.07	97.73	109.12	120.50
.02	71.26	86.34	97.73	104.39	109.12	112.78	124.16	135.53
.01	82.67	97.73	109.12	115.77	120.50	124.16	135.53	146.91

Table 22 - Sufficient Design Average Wind Tolerance (N/m)

Risk of Exceedance (%)	Designed Lifetime (years)							
	2	5	10	15	20	25	50	100
.75	3.39	5.30	6.63	7.38	7.91	8.32	9.58	10.83
.50	4.28	6.08	7.38	8.12	8.65	9.06	10.31	11.56
.40	4.73	6.50	7.79	8.53	9.06	9.46	10.72	11.97
.30	5.30	7.04	8.32	9.06	9.58	9.98	11.23	12.48
.25	5.66	7.38	8.65	9.39	9.91	10.31	11.56	12.81
.20	6.08	7.79	9.06	9.79	10.31	10.72	11.97	13.22
.15	6.63	8.32	9.58	10.31	10.83	11.23	12.48	13.73
.10	7.38	9.06	10.31	11.04	11.56	11.97	13.22	14.46
.05	8.65	10.31	11.56	12.30	12.81	13.22	14.46	15.71
.02	10.31	11.97	13.22	13.95	14.46	14.87	16.11	17.36
.01	11.56	13.22	14.46	15.19	15.71	16.11	17.36	18.61

Table 23 - Sufficient Design Gust Tolerance (N/m)

Risk of Exceedance (%)	Designed Lifetime (years)							
	2	5	10	15	20	25	50	100
.75	7.03	10.99	13.73	15.29	16.39	17.24	19.85	22.45
.50	8.86	12.61	15.29	16.84	17.93	18.77	21.37	23.97
.40	9.81	13.48	16.15	17.68	18.77	19.61	22.21	24.80
.30	10.99	14.59	17.24	18.77	19.85	20.69	23.29	25.88
.25	11.73	15.29	17.93	19.46	20.54	21.37	23.97	26.56
.20	12.61	16.15	18.77	20.29	21.37	22.21	24.80	27.39
.15	13.73	17.24	19.85	21.37	22.45	23.29	25.88	28.47
.10	15.29	18.77	21.37	22.89	23.97	24.80	27.39	29.98
.05	17.93	21.37	23.97	25.48	26.56	27.39	29.98	32.56
.02	21.37	24.80	27.39	28.90	29.98	30.81	33.40	35.98
.01	23.97	27.39	29.98	31.49	32.56	33.40	35.98	38.57

4.3 Current design standards

Rural transmission line towers are designed to similar standards throughout southern Alberta. The phase conductors (Figure 9), the wires that carry the current, all have the same failure strengths. The ground wires have a smaller failure strength, and the towers have a greater failure strength. The towers are designed so that either the insulating cable or the transmission line will break before the tower is damaged. Table 24 lists the failure loads for the transmission lines on the L-towers used in southern Alberta.

Table 24 - Load Tolerances of Transmission Lines

	Horizontal Load (N/m)	Vertical Load (N/m)
Phase wire	17.752	18.724

The tolerances in Table 24 can be compared to the tolerance in Tables 21 and 23. The vertical tolerance (Table 24) is slightly greater than the tolerance required for the line to have a 50% chance of being damaged in two years. The horizontal tolerance (Table 24) is similar to what is required for the line to have a 15% chance of being damaged in five years, a 40% chance of being damaged within fifteen years, and a 75% chance of being damaged within twenty-five years. Obviously Tables 20 and 21 overestimate the average mass and average vertical load per metre on the transmission lines. Typical transmission line spans last longer than two years. The loads listed in Tables 21 to 23 are the loads at the center of the transmission lines. The wet snow accretion is smaller near the transmission towers because the transmission lines cannot rotate near the towers. Consequently Tables 21 to 23 over estimate the average load per metre, and hence the total load that would be placed on the transmission lines.

SUMMARY AND RECOMMENDATIONS

The meteorological parameters associated with wet snow were simulated in order to study extreme wet snow accretions on transmission lines. This simulation is necessary because there are insufficient observed data, about the size and mass of wet snow accretions, to perform a risk analysis. A wet snow accretion model developed by Finstad simulated the vertical and horizontal load on a line. The model required that the air temperature, relative humidity, precipitation rate, and wind velocity be estimated throughout the event. The frequency distributions of the initial conditions for the air temperature, and wind direction during potential wet snow events have been approximated as Gaussian distributions. The frequency distributions for the bounded parameters such as relative humidity were assumed to be Gaussian only over the range of the variable. Initial conditions outside the range of the variables were ignored, and when necessary, the initial condition was re-determined until a value within the acceptable range is found. In the case of wind speed, this is equivalent to renormalizing the frequency distribution. For the visibility and the relative humidity, it is not equivalent to renormalization because the probability that the initial condition is greater than the upper limit of the range is added to probability of the occurrence of the upper limit.

The actual probability distributions, based on meteorological observations during potential wet snow

events, were not used in the model because of the ease of programming a Gaussian distribution. A Gaussian frequency distribution can be completely described by a mean and a standard deviation. Thus the initial conditions of potential wet snow events have been generated as follows; where RSD is a randomly generated number of standard deviations with a frequency distribution that is a normal distribution (a mean of zero, and a standard deviation of one):

$T_a = 2.202 + 2.49 \times RSD$ [^{o}C]

$RH = 88.856 + 9.049 \times RSD$ [%]

where RH > 100 is set to RH = 100,

and in the unlikely event that RH < 0, RH must be re-determined,

$P = 92.869 + 0.724 \times RSD$ [kPa]

$V = 11.39 + 8.751 \times RSD$ [km]

where V > 25 is set to V = 25,

and if V < 0.1, V must be re-determined,

$PR = Antilog(0.055 - \log(V) / 0.607)$ [cm snow/hr]

$U(10\ m) = 4.948 + 2.961 \times RSD$ [m/s]

where U < 0 means that U must be re-determined,

$U(36.1\ m) = 1.486 \times U(10\ m)$

$Dir = 90.0 + 80.5 \times RSD$ [degrees]

where the angle is calculated so $0 \leq Dir < 360$.

The mean wind direction was set at ninety degrees to the line because wind directions perpendicular to the line produce larger accretions than winds from other directions.

The evolution of the meteorological parameters with time was modeled in hourly steps. The temporal trends for the air temperature and the relative humidity, based on observations, are the only trends that were large enough, compared to the average standard deviations of their hourly changes, to be considered as significant. It was found that during potential wet snow events that the temperature would tend to decrease, and the relative humidity would tend to increase. These temporal trends were represented as:

$$\Delta T_{at} = -[0.156 - 0.066 \ln(t)], \text{ for } t \leq 10, \quad [^{o}C]$$
$$= 0, \text{ for } t > 10 \text{ hours } (t \text{ is in hours}),$$
$$\Delta RH_{t} = 3.172 - 1.955 \ln(t), \text{ for } t \leq 5 \text{ hours}, \quad [\%]$$
$$= 0, \text{ for } t > 5 \text{ hours}.$$

Therefore the mean hourly changes in air temperature and relative humidity are equal to the value of the temporal trend for the specific hour, while the means of the hourly changes of the other parameters are set to zero. The formulas for the frequency distributions of the hourly changes in the meteorological parameters are similar to the formulas for the frequency distributions of the initial conditions. The only major difference is that each equation for hourly changes is multiplied by a delta function.

Assuming that the distributions of the hourly changes were Gaussian, there was an unusually large fraction of observations with hourly changes equal to zero. This phenomenon was found to depend on the temperature and the change in temperature. The effects of these dependencies

were small enough to be ignored. In the extreme accretion model, the unusually large probability of an hourly changes equal to zero was simulated using a Kronecker delta function:

$\delta(x_o) = 1$ if $x > x_o$,

$= 0$ if $x \leq x_o$,

where x_o is the probability of persistence,

and x is a randomly generated number between zero and one.

The hourly changes of the parameters can be summarized as follows:

$\Delta Ta = \delta(0.34) \times (\Delta T_{at} + 0.656 \times RSD)$, [$^oC$]

$\Delta RH = \delta(0.24) \times (\Delta RH_t + 4.774 \times RSD)$, [%]

where RH > 100 is set equal to RH = 100,

and RH < 0 means that RH must be re-determined,

$\Delta P = \delta(0.17) \times 0.065 \times RSD$, [kPa]

$\Delta V = \delta(0.25) \times 6.391 \times RSD$, [km]

where V > 25 is set equal to V = 25,

and V < 0.1 means that V must be re-determined,

$\Delta U(10\ m) = \delta(0.18) \times 1.598 \times RSD$, [m/s]

where U < 0 means that U must be re-determined,

$\Delta Dir = \delta(0.39) \times 39.469 \times RSD$, [degrees]

where the angle is calculated so $0 \leq Dir < 360$.

The gust speed was estimated to exceed the wind speed by a constant times the reference velocity. For prairie conditions, an infinite plain with a roughness length of

3 cm, this constant is approximately 0.275. The assumptions that lead to a constant of proportionality are not valid where the terrain cannot be approximated as an infinite plane. Another model for gust speed will have to be developed if this extreme value model is to be used to estimate extremes for mountainous conditions.

The size of the wet snow accretion depends on the duration of the wet snow event. The cumulative probability function for the duration of a potential wet snow event, based on observations, can be described as:

$$dur_o = \text{Antilog}(-1.452 \times [(1 - P^2)^{0.5} - 1]) \text{ [hours]},$$

Where P is the cumulative probability of the duration. Based on a thermodynamical model of falling snow, the precipitation was considered to be wet snow when the surface relative humidity was greater than the critical relative humidity. Based on hourly observations and radiosonde observations, there was also a 45% chance of wet snow when the surface relative humidity was less than the critical relative humidity (RH_c), but greater than the critical surface relative humidity (RH_{cs}).

$$RH_c = 100 - 12.25\ t_a$$

$$RH_{cs} = 90.1 - 5.3\ t_a$$

These conditions were evaluated for every hour of each simulated potential wet snow event. In only a few events the precipitation will be wet snow throughout every hour of the potential wet snow event.

The number of events in a particular year influences the magnitude of the probable magnitude of the annual extreme accretion. The frequency distribution of the number of potential wet snow events in one year was approximated as Gaussian:

annual number of events = 22.632 + 7.946 x RSD.

The extreme value analysis led to the following formulas for the annual extremes as a function of the probability of exceeding these extremes:

M_{am} = -0.41 - 1.67 ln[-ln(1 - p)] [N/m]

L_{vam} = -4.23 - 16.41 ln[-ln(1 - p)] [N/m]

L_{wam} = 2.47 - 3.03 ln[-ln(1 - p)] [N/m]

L_{gam} = 3.42 - 4.18 ln[-ln(1 - p)] [N/m]

These extremes are representive of the accretion at the middle of the span. They are not representive of the average characteristics of the accretions over the whole line.

The extreme value corresponding to a specific risk and a specific mean life time can be found by combining equations (14) and (27):

$$x = u - a \ln[-\ln(1 - r) / t].$$

Therefore the minimum necessary structural strength of transmission lines, as a function of the risk and the mean life time, is the following:

L_{vam} = -4.23 - 16.41 ln[-ln(1 - r) / t], [N/m]

L_{gam} = 3.42 - 4.18 ln[-ln(1 - r) / t]. [N/m]

The risk analysis for the vertical load over-estimated the annual extreme values, while the risk analysis of the

horizontal load produced reasonable values. The differences between the quality of the two sets of data is partially due to an overestimation of the density of the accretions.

5.1 Recommendations

1. A means of estimating the mass and shape of the accretion over the entire length of the line between adjacent towers will be necessary to produce accurate estimates of accretion induced loads. Examinations of the variation of the shape and density of accretions over the length of a line are likely to require a large volume of open space. This means that wind tunnel experiments are likely to be an impractical approach to this problem. Consequently field observations of the variation of the mass and shape of wet snow accretions may be necessary.

2. The extreme accretion model developed in this study does not take line dynamics into account. It is a good model for static or near static situations, but it does not simulate the extreme loads caused by line oscillations. While a line remains intact, it may be forced to oscillate in one or more dimensions. This means that the motion of the line applies a centrifugal force (Halliday and Resnick, 1981) to the line. To examine this in detail the line oscillations would probably have to be treated as a forced, damped, simple harmonic oscillator (Marion, 1970) in one or two dimensions. This could be further complicated by considering the different types of oscillations the line might make in the process of building to its largest type of

oscillation. A study of the forcing term (gusts) and of the damping term (friction) would be necessary before good approximations could be made for the additional load due to the motion of the line. The dynamic interactions between the line and the towers should also be considered.

3. The accuracy of the mass and shape of the accretions produced by Finstad's wet snow accretion model could be improved if the accuracy of the input parameters were improved. The approximations made in this thesis for the initial values and hourly changes in these parameters (air temperature, relative humidity, precipitation rate, and wind velocity) was a first attempt at this process.

4. The estimates of the vertical loads induced by wet snow accretions would be improved if the accretion model could produce better estimates of the density, and hence the mass, of the wet snow accretions.

5. The distributions of the meteorological variables should be examined locations other than Namao, Alberta.

References

Admirat, P., J. C. Grenier, M. Maccagnan, Theory and Modelling of the Formation of Wet Snow Cylinders, Clamart Cedex, Departement Transport Appareillage, 1985a

Admirat, P., J. L. Lapeyre, M. Maccagnan, Simulation of the Cylindrical Accretion Mechanisms of Wet Snow in a Wind Tunnel, Clamart Cedex, Departement Transport Appareillage, 1985b

Finstad, Karen, Svein Fikke, Magnar Ervik, Torkild Carstens, Meteorological and Cloud Physical Observation of Atmospheric Icing Events on Gaustatoppen, 1988

Finstad, Karen J., Rime Icing Models, unpublished, 1986

Finstad, Karen, StatKaft Report, unpublished, 1989

Gumbel, E, J., Statistics of Extremes, Columbia University Press, 1958

Halliday, David, Robert Resnick,"The Dynamics of Uniform Circular Motion", Fundamentals of Physics, 2nd edition, Toronto, John Wiley & Sons, inc., 1981, pp. 84 - 86

Kinnison, Robert R., Applied Extreme Value Statistics, New York, Macmillan Publishing Company, 1985

Lozowski, E. P., K. J. Finstad, M. Bourassa, Extensions to a Wet Snow Accretion Model for Transmission Lines and Application to Alberta, research report to TransAlta Utilities, 1989

Lozowski, E. P., E. M. Gates, On The Modelling of Ice Accretion, unpublished, written in July 1987

Marion, Jerry B., Classical Dynamics of Particles and Systems, 2nd edition, New York, Academic Press, pp. 117 - 149, 1970

Matsuo, Takayo, Yoshio Sasyo, "Empirical Formula for the Melting Rate of Snowflakes", Journal of Meteorology of Japan, February, 1981a, Vol. 59, pp. 1 - 9

Matsuo, Takayo, Yoshio Sasyo, "Melting of Snowflakes below Freezing Level in the Atmosphere", Journal of Meteorology of Japan, February, 1981b, Vol. 59, pp. 10 - 25

Matsuo, Takayo, Yoshio Sasyo, "Melting of Snowflakes below Freezing Level in the Atmosphere", Journal of Meteorology of Japan, February, 1981b, Vol. 59, pp. 26 - 32

Matsuo, Takayo, Yoshio Sasyo, Yasuhiro Sato, "Relationship between Types of Precipitation on the Ground and Surface Meteorological Elements", Journal of Meteorology of Japan, 1981, Vol. 59, pp. 462 - 476

McGregor, J. R., Introduction to Statistics Notes, third printing, introductory reference notes, 1986

Stallabrass, J. R., The Airborne By Concentration of Falling Snow, DME/NAE Quarterly Bulletin, no. 1976(3), 1976

Stewart, Ronald E., "Deep 0°C Isothermal Layers Within Precipitation Bands Over Southern Ontario, Journal of Geophysical Research, vol. 89, 1984, pp. 1984

Stewart, Ronald E., "Precipitation Types in Winter Snow", PAGEOPH, vol. 123, 1985, pp. 597 - 607

Stewart, Ronald E., Patrick King, "Rain-Snow Boundaries over Southern Ontario", Monthly Weather Review, vol. 115, September, 1987, pp. 1894 - 1907

Stull, Roland B., Boundary Layer Meteorology, Boston, Kluwer Academic Publishers, 1988

Szilder, K., E. P. Lozowski, E. M. Gates, "Some Applications of a New, Time-Dependent Cylinder Ice Accretion Model", Atmospheric Research, vol. 22, Amsterdam, Elsevier Science Publishers B. V., 1988, pp. 41 - 59

Taylor, John R., An Introduction to Error Analysis, Mill Vally, University Science Books, 1982

Wakahama, Gorow, Daisuke Kuroiwa, Kazuo Goto, "Snow Accretion on Electric Wires and Its Prevention", Journal of Glaciology, 1981, Vol. 19, pp. 479 - 487

Wasserman, S. E., D. J. Monte, "A Relationship between Snow Accumulation and Snow Intensity as Determined from Visibility", Journal of Applied Meteorology, vol. 11, 1972, pp. 385 - 388

Weather Meteorological Techniques, "Precipitation", Department of the Air Force, AWS pamphlet 105-56, 1979

Appendix A - Proofs for equations in risk analysis

Proof of $T = 1 / p$

where T = the return period (expressed in numbers of trial periods)

and p = the probability of exceedance in one trial period.

Let q be the probability of non-exceedance in one trial period. Then

$$q = 1 - p.$$

Let i be the number of trial periods. Then the average number of trial periods between exceedances is:

$$T = \sum i\, p\, q^{i-1}$$

$$T = p \sum i\, q^{i-1}.$$

Substituting k for i - 1:

$$T = p \sum (k + 1)\, q^{k}$$

A formula for this summation exists on page 7 of Gradstein and Rhyzik:

$$T = p\, [(1 - q)^{-1} + q(1 - q)^{-2}]$$

$$T = 1 / p.$$

Proof of $r = 1 - (1 - p)^t$,

where r = the risk of exceedance in the time interval t,

p = the probability of exceedance in one unit of time, where the units of time must be the same as those for t,

t = the time interval of interest.

The risk is equal to the cumulative probability of exceedance over a specified number of trials (i). The number of trials will be equal to the time interval of interest (t).

$$r = p + q\,p + q^2\,p + \ldots + q^{t-1}\,p$$

$$r = p \quad q^{k-1}$$

There is a formula for this summation on page 1 of Gradstein and Rhyzik:

$$r = \frac{p\,(q^t - 1)}{q - 1}$$

$$= 1 - q^t$$

$$r = 1 - (1 - p)^t.$$

Appendix B - Melting layers

The upper air observations from Stony Plain, Alberta were examined for the presence of melting layers. Layers with a relative humidity greater than the critical relative humidity, RH_c, were considered to be melting layers. The program MELT was used to find these layers. The flowing table list the date of the upper air observations, the pressure at the bottom and the top of the layer, and the temperature and relative humidity at the bottom of the layer. Soundings with more than one melting layer are easy to identify on the table, because a date is listed only for the melting layer at the greatest altitude.

Date				Melting Layer	T_a	RH
Year	Mon	Day	Hour	(kPa)	(^oC)	(%)
67	2	6	12	No layer found		
67	4	7	24	91 to 86	4.8	32.9
67	4	15	12	No layer found		
67	4	16	24	No layer found		
67	6	9	12	92 to 79	-0.1	98.4
67	6	9	24	92 to 90	0.1	98.0
67	10	2	12	85 to 78	-0.2	96.6
				91 to 86	0.5	90.0
67	10	3	12	81 to 78	-0.4	98.3
				91 to 89	0.0	96.6
67	11	20	12	91 to 87	1.0	85.2
67	12	16	12	No layer found		
68	4	10	12	92 to 84	2.5	65.2
68	4	10	24	No layer found		
68	10	7	12	92 to 91	-0.2	100.0
69	5	14	12	92 to 83	-0.1	97.2
69	9	14	12	92 to 89	0.4	92.8
69	9	21	12	92 to 81	-0.2	98.4
69	10	2	24	91 to 82	-0.2	95.8
69	10	9	24	91 to 82	0.1	93.3
69	10	21	24	86 to 86	0.1	97.4
70	5	10	12	91 to 85	0.1	97.0
70	5	12	12	No layer found		
70	9	11	12	No layer found		
70	9	24	12	No layer found		
70	10	5	12	91 to 85	-0.4	99.0

Date				Melting Layer	T_a	RH
Year	Mon	Day	Hour	(kPa)	(°C)	(%)
70	10	9	12	No layer found		
70	10	9	24	92 to 90	1.1	69.8
70	10	12	24	92 to 91	1.7	77.4
71	1	7	24	90 to 83	0.0	100.0
71	3	19	24	No layer found		
71	10	17	24	No layer found		
71	11	11	24	86 to 85	3.0	57.0
				91 to 89	2.6	65.8
72	3	18	24	No layer found		
72	3	23	12	85 to 80	3.7	50.0
72	4	14	24	91 to 84	3.5	50.3
72	4	21	24	91 to 86	0.4	94.0
72	4	22	12	No layer found		
72	9	5	12	91 to 71	0.1	99.0
72	9	5	24	91 to 89	-0.9	100.2
72	9	19	24	91 to 83	-0.3	99.0
72	9	21	12	No layer found		
72	9	21	24	79 to 77	1.1	80.0
72	10	9	12	91 to 77	3.2	53.0
72	10	21	12	89 to 85	0.0	100.0
				92 to 92	-0.0	100.0
72	11	2	12	91 to 88	-0.1	100.0
73	2	19	24	No layer found		
73	3	10	24	No layer found		
73	9	13	12	93 to 92	0.5	91.8
73	10	6	24	91 to 87	-0.1	98.8
73	10	22	24	92 to 89	-0.0	100.0
73	10	24	24	91 to 89	-0.2	96.6
74	4	6	12	No layer found		
74	4	12	24	92 to 91	-0.0	96.9
74	5	1	24	No layer found		
74	5	9	24	91 to 84	0.3	92.0
74	5	10	12	91 to 85	0.1	95.0
74	9	9	12	79 to 75	1.3	81.0
				92 to 83	0.0	99.4
74	9	9	24	91 to 75	1.7	77.0
74	9	10	12	92 to 90	-0.4	99.0
74	10	12	24	92 to 78	2.6	64.3
75	10	7	12	92 to 92	0.7	88.1
75	10	7	24	92 to 90	-0.2	97.0
76	3	7	24	92 to 91	2.3	65.8
76	4	13	12	91 to 84	3.4	51.0
76	4	16	12	No layer found		
76	10	3	24	No layer found		
77	3	13	12	No layer found		
77	3	17	12	No layer found		
77	5	15	24	92 to 87	0.2	96.0
77	5	16	12	92 to 92	0.0	100.0
77	10	9	24	No layer found		
78	4	11	24	91 to 90	3.9	44.0
78	4	12	24	92 to 92	3.2	57.3

Date				Melting Layer	T_a	RH
Year	Mon	Day	Hour	(kPa)	(^oC)	(%)
78	4	15	24	No layer found		
78	4	17	24	92 to 87	-0.2	99.8
78	4	24	24	92 to 84	3.3	55.4
78	5	11	12	91 to 86	0.4	91.8
78	9	16	24	91 to 84	0.2	96.9
78	10	20	24	92 to 87	0.1	96.0
78	11	23	12	No layer found		
79	1	19	12	No layer found		
79	4	10	24	91 to 89	0.3	89.5
79	4	11	12	No layer found		
79	5	3	12	No layer found		
79	5	4	12	No layer found		
79	5	4	24	No layer found		
79	5	16	24	92 to 81	0.3	94.0
80	3	23	12	No layer found		
80	9	16	24	91 to 72	0.3	89.0
80	9	21	12	92 to 88	0.4	88.8
80	9	22	12	766 to 90	3.3	55.0
80	10	13	12	84 to 74	2.9	61.0
				91 to 84	3.8	48.0
80	10	13	24	92 to 85	0.0	100.0
81	10	24	24	92 to 91	0.1	91.5
82	4	8	12	No layer found		
82	5	18	24	92 to 79	0.3	94.1
83	4	3	12	92 to 91	-0.1	98.8
83	10	14	12	91 to 77	0.3	96.3
84	3	21	24	81 to 77	-0.3	97.7
				91 to 90	0.5	87.7
84	5	21	12	91 to 87	3.6	50.7
84	5	23	24	91 to 84	0.3	96.0
84	9	7	24	92 to 84	0.5	92.8
84	9	20	24	92 to 83	-0.1	99.1
84	9	21	24	92 to 87	-0.2	100.0
84	9	22	12	No layer found		
84	9	22	24	93 to 92	-0.3	97.2
84	10	17	12	No layer found		
84	10	17	24	No layer found		
84	10	18	12	No layer found		
84	10	18	24	No layer found		
84	10	19	12	No layer found		
84	10	23	24	91 to 81	-0.0	99.8
85	1	16	12	No layer found		
85	4	20	12	No layer found		
85	4	20	24	No layer found		
85	6	15	12	91 to 75	0.9	84.7
85	9	6	24	93 to 90	1.9	71.0
85	9	18	24	92 to 90	1.1	81.0
85	9	21	12	91 to 82	2.8	61.7
85	10	9	12	No layer found		
85	10	13	12	91 to 86	3.1	60.0
85	10	14	12	92 to 90	0.0	94.0

Date				Melting Layer	T_a	RH
Year	Mon	Day	Hour	(kPa)	(°C)	(%)
85	10	14	24	91 to 86	0.0	97.0
85	10	15	12	92 to 90	0.1	98.0
86	3	14	24	92 to 88	0.6	88.0
86	3	18	24	92 to 90	1.6	78.0
86	4	22	12	91 to 84	0.8	87.1
86	9	11	12	92 to 78	-0.1	94.8
86	10	1	12	93 to 86	-0.4	100.0
86	10	7	12	92 to 75	0.4	88.0

Appendix C - Temporal trends in the mean hourly changes

The following tables show the values derived from the formulas for the temporal trends of the hourly change in meteorological variables. The trend in wind direction is not listed because of the extremely poor correlation between the wind direction and time.

Time (hours)	1	2	3	4	5	6
Ta (^{o}C)	-.395	-.332	-.289	-.254	-.223	-.194
RH (%)	3.172	1.817	1.024	.462	.026	.000
Vis (km)	1.516	1.324	1.198	1.099	1.016	.943
P (kPa)	.018	.021	.023	.025	.028	.030
U (m/s)	.145	.170	.192	.212	.230	.247

Time (hours)	7	8	9	10	11	12
Ta (^{o}C)	-.166	-.137	-.105	-.063	.000	.000
RH (%)	.000	.000	.000	.000	.000	.000
Vis (km)	.876	.814	.755	.698	.642	.586
P (kPa)	.032	.034	.036	.038	.039	.000
U (m/s)	.263	.277	.292	.305	.318	.330

Time (hours)	13	14	15	16	17	18
Ta (^{o}C)	.000	.000	.000	.000	.000	.000
RH (%)	.000	.000	.000	.000	.000	.000
Vis (km)	.530	.471	.410	.342	.263	.156
P (kPa)	.000	.000	.000	.000	.000	.000
U (m/s)	.342	.354	.365	.375	.386	.396

Time (hours)	19	20	21	22	23	24
Ta (^{o}C)	.000	.000	.000	.000	.000	.000
RH (%)	.000	.000	.000	.000	.000	.000
Vis (km)	.000	.000	.000	.000	.000	.000
P (kPa)	.000	.000	.000	.000	.000	.000
U (m/s)	.406	.416	.425	.435	.444	.453

The Temporal Trend in the Hourly Change in Air Temperature

The Temporal Trend in the Hourly Change in Relative Humidity

Appendix D - Ordered Annual Extremes

The extreme values generated by the extreme accretion program are tabled in ascending order. One-hundred years of wet snow accretions were simulated. The characteristics of the transmission line were:

Diameter: 20.38 mm,

Torsional stiffness: 0.1 N/m,

Span length: 370 m.

The meteorological variable needed in the extreme accretion model were modeled upon twenty-one years of observations at CFB Namao, Alberta. Note that in two of the years there were no wet snow events.

Order	Mass (kg)	Vertical Load (N/m)	Horizontal Load (N/m)	Gust Load (N/m)
1	.000	.000	.000	.000
2	.000	.000	.000	.000
3	.000	.000	.495	.684
4	.000	.000	.745	1.028
5	.005	.045	.776	1.072
6	.005	.046	.930	1.284
7	.005	.046	.945	1.305
8	.005	.046	1.000	1.380
9	.005	.048	1.037	1.431
10	.006	.055	1.254	1.731
11	.007	.067	1.262	1.742
12	.007	.073	1.336	1.844
13	.008	.081	1.349	1.863
14	.008	.083	1.398	1.930
15	.009	.093	1.528	2.110
16	.011	.108	1.567	2.164
17	.011	.110	1.568	2.164
18	.012	.116	1.684	2.326
19	.014	.135	1.770	2.443
20	.015	.149	1.919	2.650
21	.017	.166	2.017	2.785
22	.017	.170	2.046	2.824
23	.018	.176	2.226	3.074
24	.020	.195	2.289	3.160
25	.023	.223	2.305	3.183
26	.028	.270	2.327	3.213

Order	Mass (kg)	Vertical Load (N/m)	Horizontal Load (N/m)	Gust Load (N/m)
27	.028	.272	2.378	3.284
28	.028	.273	2.442	3.372
29	.030	.298	2.472	3.413
30	.031	.306	2.582	3.565
31	.032	.314	2.667	3.683
32	.033	.326	2.687	3.709
33	.038	.370	2.689	3.713
34	.040	.389	2.745	3.790
35	.041	.403	2.799	3.864
36	.043	.418	2.822	3.896
37	.048	.468	2.845	3.928
38	.050	.492	2.962	4.090
39	.055	.535	2.970	4.101
40	.056	.550	2.993	4.132
41	.059	.580	3.073	4.243
42	.063	.617	3.220	4.446
43	.064	.627	3.287	4.539
44	.065	.636	3.339	4.610
45	.070	.690	3.347	4.621
46	.072	.708	3.401	4.696
47	.072	.709	3.410	4.709
48	.073	.717	3.474	4.797
49	.077	.760	3.534	4.879
50	.078	.764	3.591	4.958
51	.078	.769	3.640	5.026
52	.082	.801	3.803	5.250
53	.088	.860	3.887	5.368
54	.088	.866	3.911	5.400
55	.092	.907	4.010	5.537
56	.097	.950	4.101	5.662
57	.098	.965	4.119	5.687
58	.104	1.025	4.154	5.736
59	.105	1.028	4.210	5.812
60	.105	1.029	4.277	5.906
61	.105	1.033	4.282	5.912
62	.118	1.159	4.319	5.963
63	.123	1.209	4.368	6.031
64	.129	1.268	4.383	6.051
65	.134	1.319	4.420	6.103
66	.145	1.421	4.521	6.242
67	.151	1.486	4.683	6.466
68	.183	1.792	4.776	6.594
69	.201	1.976	4.843	6.687
70	.208	2.043	4.968	6.859
71	.215	2.106	4.995	6.896
72	.228	2.236	5.000	6.903
73	.241	2.365	5.264	7.268
74	.256	2.511	5.272	7.279
75	.299	2.933	5.285	7.297
76	.307	3.009	5.321	7.346
77	.310	3.040	5.427	7.493

Order	Mass (kg)	Vertical Load (N/m)	Horizontal Load (N/m)	Gust Load (N/m)
78	.312	3.062	5.558	7.675
79	.342	3.359	5.732	7.914
80	.352	3.455	5.970	8.243
81	.381	3.736	6.295	8.691
82	.420	4.122	6.359	8.780
83	.470	4.611	6.553	9.048
84	.518	5.084	6.596	9.107
85	.616	6.041	6.605	9.120
86	.627	6.155	6.624	9.147
87	.836	8.200	6.896	9.521
88	.873	8.562	7.067	9.757
89	.948	9.299	7.260	10.024
90	1.004	9.852	8.030	11.087
91	1.128	11.070	8.055	11.122
92	1.157	11.351	8.097	11.180
93	1.520	14.912	8.619	11.901
94	1.751	17.177	8.632	11.918
95	1.933	18.964	10.678	14.743
96	2.149	21.083	11.294	15.594
97	2.331	22.866	11.331	15.645
98	3.304	32.412	12.305	16.989
99	3.888	38.141	13.860	19.136
100	20.845	204.487	30.308	41.847

Appendex E - Computer programs

All the programs, except the program listed in the Section 7.5.1 meteorological variable analysis program, were developed using the Microsoft FORTRAN 4.01 compiler (a FORTRAN 77 compiler). The meteorological variable analysis program was developed using the FORTRAN 77 compiler available on the University of Alberta's Amdal computer. This program is greater than 64K, and consequently is too large to compile, without modification, on a microcomputer with an IBM (non-OS2) archetecture. The modifications made to this program, to make it work on an IBM, were to comment out the lines calling the subroutine MINIMA, and to compile the program with out the subroutine MINIMA.

7.5.1 Meteorological variable analysis program

```
C     PROGRAM METSTAT
C*****************************************************************
C                                                                *
C     Purpose: to perform a large variety of statistics          *
C           on both surface and upper air data.                  *
C           The hourly meteorological data may be                *
C           selected according to the duration of the            *
C           precipitation event or according to the              *
C           time from the onset of the event. See                *
C           the section on I/O for more details about            *
C           the options. Either one or two variables may be      *
C           examined at a time.                                  *
C              The data is arranged, in output files, so         *
C           that it can easily be plotted using                  *
C           TELL-A-GRAPH.                                        *
C                                                                *
C     definitions:                                               *
C          'number of hours from the onset of an event'          *
C          treats the hour when the precipitation event          *
C          began as the first hour                               *
C          ALLPTS - array of the particular data for the         *
C                 particular phemonena being studied. An         *
C                 index of 1 implies x-values, an index of       *
C                 2 implies y-values                             *
```

```
C         ALTHR1 - if two series of consequetive hours of  *
C                  precipitation (1, 2, 3, ..., n and m, m+1,*
C                  ..., k) are seperated by a number of    *
C                  hours (m - n), then these to precipiation *
C                  events are treated as one event of length *
C                  k, if n · ALT1 and if (m - n) μ ALTHR1   *
C         ALTHR2 - as for ALTHR1 except ALT2 replaces ALT1  *
C         ALTT1  - see ALTHR1                               *
C         ALTT2  - see ALTHR2                               *
C         BHOUR  - if data is sorted according to the time  *
C                  from the onset of the event this is the  *
C                  minimum hour of interest                 *
C         BSHAVE - if the hour at the beginning of the      *
C                  precipitation event is to be ignored then *
C                  this is equal to one, otherwise it is    *
C                  equal to zero                            *
C         CASE   - indicates how two dates compare (output  *
C                  from subroutine dtcomp):                 *
C                     0) the first date, plus the number of  *
C                     allowed hours of difference (DELHRS)   *
C                     is less than the second date,          *
C                     1) the second date, plus the number of *
C                     allowed hours of difference (DELHRS)   *
C                     is less than the first date,           *
C                     2) the first date is within the        *
C                     allowed difference of hours (DELHRS)   *
C                     of the second date                     *
C         DAY    - array containing the day of the time     *
C                  being examined. The index is the time    *
C                  from, in hours, from the onset of the    *
C                  precipitation event                      *
C         DELHRS - the minimum number of hours between      *
C                  dates, for the dates to be treated as    *
C                  the same of adjacent                     *
C         DSCRPT - character sting describing the counters  *
C                  of hours of precipitation in MNDIS: rain, *
C                  snow, rain with snow, and the total      *
C         EHOUR  - if data is sorted according to the time  *
C                  from the onset of the event this is the  *
C                  minimum hour of interest                 *
C         ESHAVE - if the hour at the end of the            *
C                  precipitation events is to be ignored    *
C                  then this is equal to one, otherwise it  *
C                  is equal to zero                         *
C         EVENTS - array counter for the frequency          *
C                  distribution for the duration of events, *
C                  or for each hour from the onset of the   *
C                  events                                   *
C         HEADNG - junk variable used to read and skip      *
C                  character headings in the data files     *
C         HOUR   - array containing the number of hours     *
C                  from the onset of the event              *
C         HOUR1  - minimum hour (from onset) of interest    *
C         HOUR2  - maximum hour (form onset) of interest    *
```

```
C         I       - counter                                          *
C         II      - counter                                          *
C         J       - counter, usually of the number of hours          *
C                   in an event                                      *
C         JFOUND  - the time (hours), from the onset of the          *
C                   precipitation event, that the upper air          *
C                   profile was observed                             *
C         JUA     - counter                                          *
C         JMIN    - the minimum duration of interest                 *
C         JMAX    - the maximum duration of interest                 *
C         K       - counter                                          *
C         LAYER   - function which returns the difference            *
C                   of a variable between the top (ULAYER)           *
C                   and the bottom (LLAYER) of a layer               *
C         LGCAVE  - logical that is true if the means                *
C                 and standard deviations of the                     *
C                 data are to be determined                          *
C         LGCDDN  - logical that is true if the                      *
C                 phemonon and its hourly change                     *
C                 are to be writen to the files                      *
C                 XDIFF.OUT and YDIFF.OUT                            *
C         LGCHIS  - logical that is true if a                        *
C                 histogram of the frequency                         *
C                 distribution is to be output to                    *
C                 the file HISTO.OUT                                 *
C         LGCMIN  - logical that is true if the best                 *
C                 fit line for the min or max                        *
C                 y-values is to be found. These                     *
C                 points will be output to the file                  *
C                 minima.out                                         *
C         LGCPCP  - logical that is ture if a listing                *
C                 of the monthly number of precip                    *
C                 events is to be writen to the                      *
C                 file PRECIP.OUT                                    *
C         LGCPRO  - logical that is true when upper                  *
C                 air profiles are being examined                    *
C         LGCRN   - logical that is true if there is                 *
C                 rain in the precipitation for the                  *
C                 hour being examined                                *
C         LGCRS   - logical that is true if there is                 *
C                 both rain and snow in the precip                   *
C                 for the hour being examined                        *
C         LGCSNW  - logical that is true if there is                 *
C                 snow in the precipitation for the                  *
C                 hour being examined                                *
C         LGCWET  - logical that is true if only wet                 *
C                 snow events are being examined.                    *
C                 this treats all hours with snow                    *
C                 and a RH greater than RHcs or RHc as wet           *
C                 snow. Wet snow events are those events             *
C                 with at least one hour of wet snow                 *
C         LGCWSN  - logical that is true if one or more              *
C                  hours of a precipitation event has wet            *
C                  snow as defined in LGCWET                         *
```

```
C         LGCXTD - logical that is true if the x          *
C              value is an hourly change                  *
C         LGCYTD - logical that is true if the y          *
C              value is an hourly change                  *
C         LGCZRO - logical that is true if hourly         *
C              changes equal to zero should not           *
C              be considered in the statistics            *
C              of the hourly changes                      *
C         LGCZXV - logical that is true if the option is  *
C               chosen to examine only the y-values       *
C               occurring when the x-value is equal to    *
C               zero.                                     *
C         LGCZXV - logical that is true if the option is  *
C               chosen to examine only the x-values       *
C               occurring when the y-value is equal to    *
C               zero.                                     *
C         LLAYER - if the change in a variable over a     *
C               layer is being examined then this is the  *
C               height of the bottom of the layer         *
C         M      - used to alter the upper limit of a     *
C               loop. The first time through the loop the *
C               the upper limit must be one less than     *
C               usual                                     *
C         MINVAL - the lower limit of the range of the    *
C               x-variable                                *
C         MNDIS  - array of the number and types of       *
C               precipitation levents sorted by year and  *
C               month of occurrance                       *
C         MONTH  - array of the month (1-12) of the time  *
C               being examined. The index is the number   *
C               of hours from the time of onset           *
C         PRESS  - array of the pressure [kPA]. The index *
C               is the number of hours from the onset of  *
C               the precipitation event                   *
C         PTS    - array of the phenomena being examined. *
C               Indices: 1) x-values                      *
C                        2) y-values                      *
C                        3) either the duration of the    *
C                        event or the time from onset     *
C         RAIN   - array of the quanity of hourly precip  *
C         RANGE  - the range of the horizontal x-variable: *
C         RH     - array of the relative humidity. The    *
C               index is the number of hours from the     *
C               onset of the event                        *
C         RUNTYP - single character choice of the group of *
C               variables to be examined (see interactive *
C               selections)                               *
C         SKIP   - if the option to use the data for every *
C               n-th hour is choosen then this is the     *
C               number of hours to be skipped (n - 1)     *
C         SLP    - array of the estimated pressure at sea *
C               level [kPa]. The index is the number of   *
C               hours from the onset of the event         *
C         SNOW   - array of the hourly intensity of       *
```

```
C                snowfall                                              *
C          SPEED - array of the wind speed. The index is               *
C                number of hours from the onset of the                 *
C                event                                                 *
C          TA    - array of hourly air temperatrue. The                *
C                index of the array is the number of hours             *
C                from the onset of the event                           *
C          TIMSET - logical that is true if events are                 *
C                sorted by their durtation, rather than by             *
C                their number of hours from the time of                *
C                the onset of the precipitation event                  *
C          TMSPAN - dummy variable for either the durataion            *
C                of the event, or the number of hours from             *
C                the onset of the event
C          TW    - array of wet bulb temperatures. The                 *
C                index is the number of hours from the                 *
C                onset of the event.
C          UA    - logical that is true if upper air                   *
C               data will be used                                      *
C          UACNT - counter for the number of lines on one              *
C                page of upper air data                                *
C          UACASE - as for case except the first date is a             *
C                date of an upper air sounding, and the                *
C                second date is a date of a surface event.             *
C                DELHRS is zero.                                       *
C          UADATA - array of upper air data. The first                 *
C                index (JUA) is a counter of the number                *
C                of layers. The second index is for the                *
C                type (1-15) of the phenomena:                         *
C                   1) pressure [kPa] at the top of a layer*
C                   2) altitude [m],                                   *
C                   3) air temperature [degrees Celcius]               *
C                   4) relative humidity [%]                           *
C                   5) wind direction [degrees]                        *
C                   6) wind speed [m/s]                                *
C                   7 - 12) as for (1 to 6) for the bottom             *
C                   of the layer                                       *
C                   13) depth of the layer [m]                         *
C                   14) lapse rate of the layer                        *
C                   15) wind shear in the layer                        *
C          UADATE - array for the time of the upper air                *
C                profile: year, month, day, hour                       *
C          UAFND - logical that is true if upper air                   *
C               data is found for the time of                          *
C               interest                                               *
C          UAVAL - a dummy variable for the upper air                  *
C                phenomenon being examined                             *
C          ULAYER - if the value of a variable at a                    *
C                particular height (or difference of                   *
C                heights) this is the height (or the                   *
C                height of the top of the layer                        *
C          VIS   - array of the visibility [km]. The index             *
C                is the number of hours from the onset of              *
C                the precipitation event                               *
```

```
C          WIDTH  - the size (width) of data pools, on the  *
C                   (horizontal) x-axis. Used only in finding *
C                   the best fit line for either minima or    *
C                   maxima of selected of a phenomena         *
C                   associated with selected types of         *
C                   precipitation                             *
C          WINDIR - array of the wind direction [degrees].   *
C                   The index is the number of hours from the *
C                   onset of the event                        *
C          X      - character string describing the          *
C                   x-variable                                *
C          XMAX   - the maximum (horizontal) x-value         *
C          XMIN   - the minimum (horizontal) x-value         *
C          XVAL   - array of the (horizontal) values of the  *
C                   x-phenomenon                              *
C          XVALUE - single character choice of the           *
C                   x-variable                                *
C          Y      - character string describing the          *
C                   y-variable                                *
C          YEAR   - array containing the year of the time    *
C                   being examined. The index is the number   *
C                   hours from the onset of the event         *
C          YEARN  - the last year for which there is data    *
C          YEAR1  - the earliest year for which there is     *
C                   surface data                              *
C          YMAX   - maximum (vertical) y-value               *
C          YMIN   - minimum (vertical) y-value               *
C          YRNDEX - a dummy index equal to the year of the   *
C                   data being examined, minus the first      *
C                   year, plus one.                           *
C          YVAL   - array of the (vertical) values of the    *
C                   y-phenomenon being examined               *
C          YVALUE - single character choice of the           *
C                   y-variable                                *
C                                                             *
C     I/O Streams                                            *
C          4 file RHVTA.BAT contains the input for the       *
C              interactive routine SPEAK. The I/O streams    *
C              can be made truely interactive by changing    *
C              the stream 4 of the reads in SPEAK.FOR to     *
C              stream 5.                                     *
C          5 keyboard                                        *
C          6 screen                                          *
C          7 file SFC.DAT the file of surface data. From     *
C              left to right, the data contained in this     *
C              file must be: hour, day, month, year,         *
C              hourly rainfall, hourly intensity of          *
C              snowfall, air temperature [C], relative       *
C              humidity [%], dew point temperature [C],      *
C              wet bulb temperature [C], wind speed [m/s],   *
C              wind direction (degress, 0=N,90=E),           *
C              pressure [10 kPa], equivelent pressure at     *
C              sea level [kPa], visibility [km]. These       *
C              must match to format number 9002              *
```

```
C          8 file RAIN.OUT - stores the x and y variables   *
C            from hours when rain occurred, but snow did *
C            not occur                                   *
C          9 file SNOW.OUT - stores the x and y variables   *
C            from hours when snow occurred, but rain did *
C            not occur                                   *
C         10 file RNSW.OUT - stores the x and y varialbes   *
C            from hours when both snow and rain occurred *
C         11 file RAIN2.OUT - as for rain.out, except it is *
C            only used when the beginning, middle and    *
C            end hours of precipitation events are being *
C            plotted. The files RAIN.OUT, SNOW.OUT, and  *
C            RNSNW.OUT are used for the beginning hours. *
C            RAIN2.OUT, SNOW2.OUT, and RNSW2.OUT are     *
C            used for the middle hours. Files RAIN3.OUT, *
C            SNOW3.OUT, and RNSW3.OUT are used for the   *
C            end hours                                   *
C         12 file SNOW2.OUT - see I/O stream 11           *
C         13 file RNSW2.OUT - see I/O stream 11           *
C         14 file RAIN3.OUT - see I/O stream 11           *
C         15 file SNOW3.OUT - see I/O stream 11           *
C         16 file RNSW3.OUT - see I/O stream 11           *
C         17 file TITLE.OUT - the titles for a TELL-A-GRAPH *
C            plot is output here                         *
C         18 file STATS.OUT - statistics from the least    *
C            squares analysis of the minimum (or         *
C            maximum) points in each pool are output     *
C            here                                        *
C         19 file MINIMA.OUT - the minima used in least    *
C            squares analysis of the minimum (or         *
C            maximum) points in each pool are output     *
C            here                                        *
C         20 file UA.DAT - data file for the upper air     *
C            profiles. The oder of the data (from left   *
C            to right) must be: year, month, day, hour;  *
C            six data from the top of the layer:         *
C            pressure [kPA], altitude [m], air           *
C            temperature [C], relative humidity [%],     *
C            wind direction [degrees, 0=N], and wind     *
C            speed (m/s); the same six varaibles from    *
C            bottom of the layer; thickness of the layer *
C            [m], lapse rate [C/km], and the wind shear  *
C            [1/C]                                       *
C         21 file PRECIP.OUT - output of the number of     *
C            hours and totals of each type of            *
C            precipitation for each month of each year   *
C         22 file STATS2.OUT - output of a least squares   *
C            analysis of all x and y data, or an         *
C            analysis of the mean and standard deviation *
C            if only one variable is being examined.     *
C         23 file ZEROS.OUT - if the values of the         *
C            variables that are equal to zero are to be  *
C            ignored in the statistical analysis then    *
C            data on their frequency is output here      *
```

```
C        24 file HISTO.OUT - if only one variable is being *
C             examined, and if a histogram is requested,  *
C             then the data for the histogram is output   *
C             here                                        *
C        25 file XDIF.OUT - if the x-values and the hourly *
C             change in the x-values are requested, then  *
C             this is where they will be output           *
C        26 file YDIF.OUT - if the y-values and the hourly *
C             change in the y-values are requested, then  *
C             this is where they will be output           *
C                                                         *
C         Programmed by Mark Bourassa                     *
C         September, 1988                                 *
C         University of Alberta                           *
C                                                         *
C*********************************************************

      LOGICAL LGCRN, LGCSNW, LGCRS, LGCMIN, LGCAVE, LGCPCP,
     + TIMSET, UA, UAFND, LGCZRO, LGCHIS, LGCDDN, LGCXTD,
     + LGCYTD, LGCPRO, LGCWET, LGCWSN, LGCZXV, LGCZYV
      INTEGER I, J, MONTH(60), DAY(60), HOUR(60), START,
     + DELHRS, YEAR(60), K, M, JMIN, JMAX, BSHAVE, ESHAVE,
     + SKIP, EVENTS(62), TMSPAN, HOUR1, HOUR2, EHOUR, BHOUR,
     + II, CASE, JUA, UADATE(4), UAVAL, UACASE, JFOUND,
     + UACNT, MNDIS(4,30,12), YEAR1, YEARN, ALLPTS(2),
     + ALTT1, ALTT2, ALTHR1, ALTHR2, YRNDEX
      REAL RAIN(60), SNOW(60), TA(60), RH(60), TW(60),
     + RANGE, XVAL(60), YVAL(60), WIDTH, MINVAL, ULAYER,
     + LLAYER, UADATA(30,15), LAYER, SPEED(60), WINDIR(60),
     + PRESS(60), SLP(60), XMIN, XMAX, YMIN, YMAX,
     + PTS(1600,3), VIS(60)
      CHARACTER X*7, Y*10
      CHARACTER*1 XVALUE, YVALUE, HEADNG, RUNTYP
      CHARACTER*5 DSCRPT(4)

C     open the I/O streams
      OPEN (UNIT = 7, FILE = 'SFC.DAT', STATUS = 'UNKNOWN' )
      OPEN (UNIT = 8, FILE = 'RAIN.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT= 9, FILE = 'SNOW.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=10, FILE = 'RNSW.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=11, FILE= 'RAIN2.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=12, FILE= 'SNOW2.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=13, FILE= 'RNSW2.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=14, FILE= 'RAIN3.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=15, FILE= 'SNOW3.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=16, FILE= 'RNSW3.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=17, FILE= 'TITLE.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=18, FILE= 'STATS.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=19, FILE= 'MINIMA.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=20, FILE= 'UA.DAT', STATUS = 'UNKNOWN' )
      OPEN (UNIT=21, FILE= 'PRECIP.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=22, FILE= 'STATS2.OUT', STATUS = 'UNKNOWN')
      OPEN (UNIT=23, FILE= 'ZEROS.OUT', STATUS = 'UNKNOWN' )
      OPEN (UNIT=24, FILE= 'HISTO.OUT', STATUS = 'UNKNOWN' )
```

```
      OPEN (UNIT=25, FILE= 'XDIF.OUT', STATUS = 'UNKNOWN' )
      OPEN (UNIT=26, FILE= 'YDIF.OUT', STATUS = 'UNKNOWN' )

C     initialize vairables
      DATA EVENTS/62*0/
      DATA MNDIS/1440*0/
      DATA ALLPTS/2*0/
      LGCWSN = .FALSE.
      CASE = 1
      JUA = 1
      M = 0
      J = 1
      DELHRS = 1
      DSCRPT(1) = 'RAIN '
      DSCRPT(2) = 'SNOW '
      DSCRPT(3) = 'BOTH '
      DSCRPT(4) = 'TOTAL'
      UAFND = .FALSE.
      UACASE = 1
      UACNT = 1
      XMIN = 99999.0
      XMAX = -99999.0
      YMIN = 99999.0
      YMAX = -99999.0

C     skip over the character headings at the top of the
C     page of surface data
      READ(7,9001) HEADNG
      READ(7,9001) HEADNG

C     call the interactive routine
      CALL SPEAK(LGCRN,LGCSNW,LGCRS,LGCMIN,LGCAVE,LGCPCP,
     2 START,JMIN,JMAX,BSHAVE,ESHAVE,SKIP,DPTS,WIDTH,X,Y,
     3 XVALUE, YVALUE,RUNTYP,S,RANGE,MINVAL,TIMSET,BHOUR,
     4 EHOUR,ULAYER, LLAYER,UA,ALTT1,ALTT2,ALTHR1,ALTHR2,
     5 LGCZRO,LGCHIS,LGCNRM, LGCDDN,LGCXTD,LGCYTD,LGCPRO,
     6 LGCWET,LGCZXV,LGCZYV)

C     read the surface data
   50 READ(7,9002,END=4000) HOUR(J), DAY(J), MONTH(J),
     2 YEAR(J), RAIN(J), SNOW(J), TA(J), RH(J), TW(J),
     3 SPEED(J), WINDIR(J), PRESS(J), SLP(J), VIS(J)
C     convert pressure to kPa
      PRESS(J) = PRESS(J) / 10.0
C     set the earliest year
      YEAR1 = YEAR(J)
C     if upper are data is going to be used
      IF ( UA ) THEN
C     read (and ignore) the three lines of headings
         DO 65 K=1, 3
            READ(20,9001) HEADNG
   65    CONTINUE
C     read the time at which the upper air profile was
C     observed, and the data for the top layer in the profile
```

```
            READ(20,9015,END=4000) (UADATE(K),K=1,4),
     2     (UADATA(JUA,K),K=1,15)
C       set the counter for the next line numbe,r in the upper
C       air file, at five
            UACNT = 5
          ENDIF
          J = 2
C       continue reading and analyzing surface data until the
C       end of a page (60 lines) is reached.
  100   DO 500 I=1,57+M
            READ(7,9002,END=4000) HOUR(J), DAY(J), MONTH(J),
     2     YEAR(J), RAIN(J), SNOW(J), TA(J), RH(J), TW(J),
     3     SPEED(J), WINDIR(J), PRESS(J), SLP(J), VIS(J)
            PRESS(J) = PRESS(J) / 10.0
C       compare the time of occurance of the last two hours of
C       precipitation.
            CALL DTCOMP(YEAR(J-1),MONTH(J-1),DAY(J-1),
     2     HOUR(J-1), YEAR(J),MONTH(J),DAY(J),HOUR(J),CASE,
     3     DELHRS)
C       if the last two hours are adjacent in time then
C       increment the counter of the number of hours in
C       an event.
            IF ( CASE .EQ. 0 ) THEN
               J = J + 1
C       if there has been enough hours in the event then lulls
C       of one or more hours might (optionally) be allowed to
C       occur within events. If so then the minimum difference
C       in time between hourly precipitation is increased from
C       one to the alternate value.
               IF ( J .GT. ALTT1 ) DELHRS = ALTHR1
               IF ( J .GT. ALTT2 ) DELHRS = ALTHR2
C       if the last to two hours are not considered part of the
C       same precipitation event, and if some of the duration
C       is within the range of durations of interest, then
C       these hours are analyzed.
            ELSEIF ((J-1 .GE. JMIN) .AND. (J-1 .LE. JMAX)) THEN
C       if upper air data is required then each hour of the
C       event is tested to determine if there is an upper air
C       profile for the same time.
               IF ( UA .OR. LGCPRO ) THEN
                  II = BSHAVE
  195             IF ( II .LE. J-ESHAVE ) THEN
C       compare the dates, moving through the upper air data,
C       until the upper air date is equal to or greater than
C       the surface date. If the dates are equal then set UAFND
C       to be true.
                     CALL UATEST( JUA, YEAR, MONTH, DAY, HOUR,
     2              UADATE, UADATA, UACNT, UAFND, JFOUND, II )
C       if the dates did not match then increment try the next
C       hour of the precipitation event.
                     IF ( .NOT. UAFND ) THEN
                        II = II + 1
                        GOTO 195
                     ELSE
```

```
C     if the dates matched, then set the lower (HOUR1) and
C     upper (HOUR2) limits for the times of interest
                    IF ( UA ) THEN
C     if the upper air profile is being examined then only
C     the hour when the profile was observed is of interest.
                      HOUR1 = JFOUND
                      HOUR2 = JFOUND
                    ELSEIF ( TIMSET ) THEN
C     if the data is being sorted according to the duration
C     of the event then tests have already been preformed to
C     confirm that the duration of this event is in the range
C     of interest. If the first or the last hours of the
C     event are (optionally) to be ignored, then the range of
C     hours is adjusted
                      HOUR1 = BSHAVE
                      HOUR2 = J - ESHAVE
                    ELSE
C     if the data is being sorted according to the time from
C     the onset of the event (and optionally the duration)
C     then the range of hours of interest is set.
                      HOUR1 = BHOUR
                      HOUR2 = EHOUR
                      IF ( HOUR1 .LT. BSHAVE )
     2                HOUR1 = BSHAVE
                      IF ( HOUR2 .GT. J - ESHAVE ) HOUR2 =
     2                J - ESHAVE
                    ENDIF
C     subroutine SETVAL determines the (horixontal) x-values
                    CALL SETVAL( XVALUE, XVAL, TA, TW,
     2              UADATA, ULAYER, LLAYER, J, JUA, HOUR1,
     3              HOUR2, RH, SPEED, WINDIR, PRESS, VIS,
     4              XMIN, XMAX, .TRUE., LGCDDN, LGCXTD,
     5              LGCZXV, RAIN, SNOW )
C     subroutine SETVAL determines the (vertical) y-values
                    CALL SETVAL( YVALUE, YVAL, TA, TW,
     2              UADATA, ULAYER, LLAYER, J, JUA, HOUR1,
     3              HOUR2, RH, SPEED, WINDIR, PRESS, VIS,
     4              YMIN, YMAX, .FALSE., LGCDDN, LGCYTD,
     5              LGCZYV, RAIN, SNOW )
C     if the hourly changes in either x or y are being
C     examined, then then the number of variables is reduced
C     by one. To compensate for this the range of hours is
C     reduced by one
                    IF ( LGCXTD .OR. LGCYTD ) HOUR2 =
     2              HOUR2 - 1
C     subroutine IDENT selects the variables from the times
C     of interest, and copies them into the array PTS
                    CALL IDENT( TIMSET, UA, RUNTYP, HOUR1,
     2              HOUR2, BSHAVE, ESHAVE, J, SKIP, RAIN,
     3              SNOW, XVAL, YVAL, RANGE, MINVAL,
     4              LGCSNW, LGCRN, LGCRS, JMIN, ALLPTS,
     5              PTS, LGCAVE )
C     if the option to examine only wet snow events is not
C     being used, then subroutine DSTRBN counts the hours of
```

```
C     rain, snow, and rain with snow for each month of each
C     year.
                        IF ( .NOT. LGCWET )
     2                  CALL DSTRBN( TIMSET, UA, LGCRN, LGCSNW,
     3                  LGCRS, EVENTS, HOUR1, HOUR2, MNDIS,
     4                  YEAR, MONTH, YRNDEX, YEAR1, SNOW, RAIN)
                        IF ( LGCXTD .OR. LGCYTD ) HOUR2 =
     2                  HOUR2 + 1
C     reset UAFND to false - the precipitation evetn may be
C     long enough to have more than one hour match that of an
C     upper air sounding
                        UAFND = .FALSE.
C     the last (first index) set of upper air data, in the
C     array of upper air data, is the first set of data in
C     the next profile. This data is moved to the first
C     postion (JUA = 1).
                        DO 197 K=1, 15
                           UADATA(1,K) = UADATA(JUA,K)
   197                  CONTINUE
C     repeat this until all hours of the surface precipiation
C     event have been examined
                        II = II + 1
                        GOTO 195
                     ENDIF
                  ENDIF
C     if no upper air data is required then the lower (HOUR1)
C     and  upper (HOUR2) limits for the times of interest are
C     set
               ELSE
                  IF ( TIMSET ) THEN
C     if the data is being sorted according to the duration
C     of the event then tests have already been preformed to
C     confirm that the duration of this event is in the range
C     of interest. If the first or the last hours of the
C     event are (optionally) to be ignored, then the range of
C     hours is adjusted
                     HOUR1 = BSHAVE
                     HOUR2 = J - ESHAVE
                  ELSE
C     if the data is being sorted according to the time from
C     the onset of the event (and optionally the duration)
C     then the range of hours of interest is set.
                     HOUR1 = BHOUR
                     HOUR2 = EHOUR
                     IF ( HOUR1 .LT. BSHAVE ) HOUR1 = BSHAVE
                     IF ( HOUR2 .GT. J - ESHAVE ) HOUR2 =
     2               J - ESHAVE
                  ENDIF
C     if the option to examine only the data from
C     precipitation events where wet snow occurred is used,
C     then each hour of the event is examined to determine if
C     wet snow was resonably likely (45% or greater) to have
C     fallen. If the RH is greater than RHc or RHcs this is
C     true.
```

```
                IF ( LGCWET ) THEN
                   LGCWSN = .FALSE.
                   DO 300 II=HOUR1, HOUR2
                      IF ( ( SNOW(II) .GT. 0.0 ) .AND.
     2                ( (RH(II) .GE. 100.0 - 12.25 * TA(II) )
     3                .OR. (RH(II) .GE. 90.1 - 5.3 * TA(II) )
     4                ) ) LGCWSN = .TRUE.
  300              CONTINUE
                ENDIF
C     the data from the event is examined, unless only wet
C     snow events are being examined, and there was no wet
C     snow.
                IF ( ( .NOT. LGCWET ) .OR. LGCWSN ) THEN
C     subroutine SETVAL determines the (horixontal) x-values
                   CALL SETVAL( XVALUE, XVAL, TA, TW, UADATA,
     2             ULAYER, LLAYER, J, JUA, HOUR1, HOUR2, RH,
     3             SPEED, WINDIR, PRESS, VIS, XMIN, XMAX,
     4             .TRUE., LGCDDN, LGCXTD, LGCZXV, RAIN,SNOW)
C     subroutine SETVAL determines the (horixontal) x-values
                   CALL SETVAL( YVALUE, YVAL, TA, TW, UADATA,
     2             ULAYER, LLAYER, J, JUA, HOUR1, HOUR2, RH,
     3             SPEED, WINDIR, PRESS, VIS, YMIN, YMAX,
     4             .FALSE., LGCDDN, LGCYTD, LGCZYV,RAIN,SNOW)
C     if the hourly changes in either x or y are being
C     examined, then then the number of variables is reduced
C     by one. To compensate for this the range of hours is
C     reduced by one
                   IF (LGCXTD .OR. LGCYTD) HOUR2 = HOUR2 - 1
C     subroutine IDENT selects the variables from the times
C     of interest, and copies them into the array PTS
                   CALL IDENT( TIMSET, UA, RUNTYP, HOUR1,
     2             HOUR2, BSHAVE, ESHAVE, J, SKIP, RAIN,
     3             SNOW, XVAL, YVAL, RANGE, MINVAL, LGCSNW,
     4             LGCRN, LGCRS, JMIN, ALLPTS, PTS, LGCAVE )
                   CALL DSTRBN( TIMSET, UA, LGCRN, LGCSNW,
     2             LGCRS, EVENTS, HOUR1, HOUR2, MNDIS, YEAR,
     3             MONTH, YRNDEX, YEAR1, SNOW, RAIN )
                   IF (LGCXTD .OR. LGCYTD) HOUR2 = HOUR2 + 1
                ENDIF
             ENDIF
C     the last values in the arrays of surface data are the
C     first values of the next precipitation event. After the
C     event is examined, the first values of the arrays are
C     reset to equal the first values of the next
C     precipitation event.
             YEAR(1) = YEAR(J)
             MONTH(1) = MONTH(J)
             DAY(1) = DAY(J)
             HOUR(1) = HOUR(J)
             TA(1) = TA(J)
             TW(1) = TW(J)
             RH(1) = RH(J)
             RAIN(1) = RAIN(J)
             SNOW(1) = SNOW(J)
```

```
            SPEED(1) = SPEED(J)
            WINDIR(1) = WINDIR(J)
            PRESS(1) = PRESS(J)
            SLP(1) = SLP(J)
            VIS(1) = VIS(J)
            J = 2
C     the maximum difference between hours is reset to one
            DELHRS = 1
          ELSE
C     even if the event was not examined, because the
C     duration was outside the range of interest, the first
C     values of the surface data arrays must be reset to be
C     equal to the first values for the next event.
            YEAR(1) = YEAR(J)
            MONTH(1) = MONTH(J)
            DAY(1) = DAY(J)
            HOUR(1) = HOUR(J)
            TA(1) = TA(J)
            TW(1) = TW(J)
            RH(1) = RH(J)
            RAIN(1) = RAIN(J)
            SNOW(1) = SNOW(J)
            SPEED(1) = SPEED(J)
            WINDIR(1) = WINDIR(J)
            PRESS(1) = PRESS(J)
            SLP(1) = SLP(J)
            VIS(1) = VIS(J)
C     if the duration of the event was greater than the size
C     of the data array, a warning is written to the screen
            IF ( J .GT. 60 ) WRITE(6,9006) 'WARNING',
     2      ' DURATION OF ', J
            J = 2
          ENDIF
  500 CONTINUE
C     after each page of surface data read the character
C     headings from the top of the next page
       READ(7,9001,END=4000) HEADNG
       READ(7,9001,END=4000) HEADNG
C     after the first loop, change the upper limit of the
C     loop for the number of lines of surface data on a page
C     to be 58 (57 + 1).
       M=1
       GOTO 100

C     set the last year for which there was dataf
 4000 YEARN = YEAR(J-1)

C     output (to the screen) the distribution of the duration
C     or the number of hours with precipitation at each hour
C     from the time of onset
       DO 4100 I=0,5
          WRITE(6,9013) (K,K=10*I+1,10*(I+1))
          WRITE(6,9013) (EVENTS(K),K=10*I+1,10*(I+1))
          WRITE(6,9003) ' '
```

```
 4100 CONTINUE
      WRITE(6,9014) 'TOTAL NUMBER OF EVENTS IS ', EVENTS(61)
      WRITE(6,9014) 'TOTAL NUMBER OF HOURS IS ', EVENTS(62)

C     if the option for the a count of the number of hours
C     with each type of precipitation was chosen then output
C     this information to the file PRECIP.OUT.
      IF ( LGCPCP .AND. ( .NOT. UA ) ) THEN
         DO 5200 II=YEAR1, YEARN
            J = II - YEAR1 + 1
            WRITE(21,9003) ' '
            WRITE(21,9003) 'YEAR                ',
     2      '                    MONTH'
            WRITE(21,9016) (K,K=1,12)
            DO 5100 I=1,4
               WRITE(21,9017) II, DSCRPT(I),
     2         (MNDIS(I,J,K),K=1,12)
 5100       CONTINUE
 5200    CONTINUE
      ENDIF

C     if the option for general statistics was chosen then
C     set the (hourly) limits for the data of interest, and
C     call the statistics subroutine AVERAG
 9000 IF ( TIMSET ) THEN
         HOUR1 = JMIN
         HOUR2 = JMAX
      ELSE
         HOUR1 = BHOUR
         HOUR2 = EHOUR
         IF ( HOUR1 .LT. BSHAVE ) HOUR1 = BSHAVE
         IF ( HOUR2 .GT. J - ESHAVE ) HOUR2 = J - ESHAVE
      ENDIF
      IF ( LGCAVE .AND. ( EVENTS(62) .GT. 2 ) ) THEN
         IF ( LGCZXV ) XVALUE = 'Z'
         IF ( LGCZYV ) YVALUE = 'Z'
         CALL AVERAG( HOUR1, HOUR2, ALLPTS(1), PTS, XVALUE,
     2   YVALUE, LGCZRO, LGCHIS, LGCNRM )
      ENDIF

 9001 FORMAT(A)
 9002 FORMAT(2X,3(I2,3X),1X,I2,2X,F8.1,10X,3(2X,F8.1),12X,
     2 F8.1,F8.2,4(F8.1,2X))
 9003 FORMAT(1X,A,A)
 9004 FORMAT(I2)
 9005 FORMAT(1X,F8.2,A,F8.2)
 9006 FORMAT(1X,A,A,I2,A)
 9007 FORMAT(1X,A,I2,A,I2,1X,A)
 9008 FORMAT(1X,A,A,A,A,A)
 9009 FORMAT(F4.2)
 9010 FORMAT( 1X,A,F12.4,A,F12.4,A )
 9013 FORMAT(1X,10(I3,1X))
 9014 FORMAT(1X,A,I4)
 9015 FORMAT(1X,3(I2,1X),I2,F6.2,F7.0,F7.1,3F7.0,F7.2,F7.0,
```

```
     2 F7.1,4F7.0,3F7.2)
 9016 FORMAT(13X,12I4)
 9017 FORMAT(2X,I2,1X,A,3X,12I4)

C    end of main program METSTAT
      END
```

7.5.1.1 Function UAPT

```
C FUNCTION UAPT ************************************************
C                                                               *
C    Purpose: to determine the value of one varaible at a       *
C         specific pressure in atmosphere. There is no          *
C         extrapolation.                                        *
C                                                               *
C    Definitions:                                               *
C         FUDGE  - multiplier to move the decimal of the        *
C                  error designator (-9999.9), so that it       *
C                  matches the error designator of the          *
C                  variable                                     *
C         JUA    - the number of heights at which              *
C                  observations were made                       *
C         L      - index (counter) for the hieght in the       *
C                  atmosphere. Starts at the top of the         *
C                  layer                                        *
C         UADATA - see METSTAT                                  *
C         UAPT   - the returned value                           *
C         UAVAL  - an index of UADATA that indicates the        *
C                  variable of interest                         *
C         ULAYER - the height of interest                       *
C                                                               *
C    Programmed by Mark Bourassa                                *
C                                                               *
C***************************************************************

      FUNCTION UAPT(UADATA,ULAYER,JUA,UAVAL)
      INTEGER JUA, UAVAL, L
      REAL UADATA(30,15), ULAYER, FUDGE, UAPT

      L = 1
      FUDGE = 1.0
      IF ( UAVAL .EQ. 1 ) FUDGE = 100.0
      IF ( UAVAL .EQ. 3 ) FUDGE = 10.0

      IF ( ULAYER .LT. 0.0 ) THEN
         UAPT = UADATA(JUA-1,UAVAL)
      ELSE
  100    IF ( UADATA(L,1) .EQ. ULAYER ) THEN
            IF (UADATA(L,UAVAL) .EQ. -99999.0 / FUDGE) THEN
               UAPT = -99999.0
            ELSE
               UAPT = UADATA(L,UAVAL)
            ENDIF
```

```
         ELSEIF ( L .LT. JUA - 1 ) THEN
            L = L + 1
            GOTO 100
         ELSE
            UAPT = -99999.0
         ENDIF
       ENDIF
       RETURN
C    end of function UAPT
       END
```

7.5.1.2 Function LAYER

```
C  FUNCTION LAYER  ******************************************
C                                                            *
C    Purpose: to determine the difference of a variable      *
C         over a layer in the atmosphere. Oservations must   *
C         have been made at these pressures for a values     *
C         to be returned, otherwise -9999.9 is returned.     *
C                                                            *
C    Definitions:                                            *
C         FUDGE  - multiplier to move the decimal of the     *
C                  error designator (-9999.9), so that it    *
C                  matches the error designator of the       *
C                  variable                                  *
C         JUA    - the number of heights at which           *
C                  observations were made                    *
C         L      - index (counter) for the hieght in the    *
C                  atmosphere. Starts at the top of the      *
C                  layer                                     *
C         LAYER  - the returned value                       *
C         LLAYER - the pressure at the bottom of the layer  *
C         TOPJUA - index for the pressure at the top of     *
C                  the layer: ULAYER = UADATA(TOPJUA,1)      *
C         UADATA - see METSTAT                              *
C         UAVAL  - an index of UADATA that indicates the    *
C                  variable of interest                      *
C         ULAYER - the pressure at the top of the layer     *
C                                                            *
C    Programmed by Mark Bourassa                             *
C                                                            *
C************************************************************
```

```
      FUNCTION LAYER(UADATA,ULAYER,LLAYER,JUA,UAVAL)
      INTEGER JUA, UAVAL, L, TOPJUA
      REAL UADATA(30,15), ULAYER, LLAYER, FUDGE, LAYER

      L = 1
      FUDGE = 1.0
      IF ( UAVAL .EQ. 1 ) FUDGE = 100.0
      IF ( UAVAL .EQ. 3 ) FUDGE = 10.0
```

```
100 IF ( UADATA(L,1) .EQ. ULAYER ) THEN
       TOPJUA = L
       L = L + 1
200    IF ( ( UADATA(L,1) .EQ. LLAYER ) .OR.
   2   ( LLAYER .LT. 0.0 ) ) THEN
          IF ( LLAYER .LT. 0.0 ) L = JUA - 1
          IF ( (UADATA(TOPJUA,UVAL) .EQ. -99999.0 / FUDGE)
   2      .OR. ( UADATA(L,UVAL) .EQ. -99999.0 / FUDGE ) )
   3      THEN
             LAYER = -99999.0
          ELSE
             LAYER =UADATA(TOPJUA,UAVAL) - UADATA(L,UAVAL)
          ENDIF
       ELSEIF ( L .LT. JUA - 1 ) THEN
          L = L + 1
          GOTO 200
       ELSE
          LAYER = -99999.0
       ENDIF
    ELSEIF ( L .LT. JUA - 1 ) THEN
       L = L + 1
       GOTO 100
    ELSE
       LAYER = -99999.0
    ENDIF
    RETURN
C   end of function LAYER
    END
```

7.5.1.3 Function UATEST

```
C  SUBROUTINE UATEST  ****************************************
C                                                             *
C    Purpose: to find a upper air sounding that was           *
C         observed on the same time (hour) as a surface       *
C         precipitation event occurred. It searches for       *
C         the upper air date until the date of the            *
C         sounding is later than or equal the date of the     *
C         profile.                                            *
C                                                             *
C    Definitions:                                             *
C         DAY     - day of the surface precipitaiton event    *
C         HOUR    - hour of the surface precipitation event   *
C         II      - ses METSTAT                               *
C         JFOUND - the number of hours from the onset of      *
C                  precipitation event                        *
C         JUA     - ses METSTAT                               *
C         K       - counter                                   *
C         MONTH   - month of the surface precipitation        *
C                  event                                      *
C         UACASE - ses METSTAT                                *
C         UACNT   - ses METSTAT                               *
C         UADATA - ses METSTAT                                *
```

```
C          UADATE - see METSTAT                                  *
C          UAFND  - ses METSTAT                                  *
C          YEAR   - year of the surface event                    *
C                                                                *
C     Programmed by Mark Bourassa   1988                         *
C                                                                *
C****************************************************************

       SUBROUTINE UATEST( JUA, YEAR, MONTH, DAY, HOUR,
      2 UADATE, UADATA, UACNT, UAFND, JFOUND, II )

       LOGICAL UAFND
       INTEGER JUA, YEAR(60), MONTH(60), DAY(60), HOUR(60),
      2 UACASE, UADATE(4), UACNT, K, II, JFOUND
       REAL UADATA(30,15)
       CHARACTER*1 HEADNG

       JUA = 1
C      compare the surface and sounding dates
  155  CALL DTCOMP(UADATE(1),UADATE(2),UADATE(3),UADATE(4),
      2 YEAR(II),MONTH(II),DAY(II),HOUR(II),UACASE,0)
C      if the sounding date is earlier than the surface date,
C      then read in the next upper air layer, and compare
C      dates again
       IF ( UACASE .EQ. 1 ) THEN
C      skip over the character headings at the top of each
C      page
          IF ( UACNT .EQ. 1 ) THEN
             DO 160 K=1, 3
               READ(20,9001) HEADNG
  160        CONTINUE
             UACNT = 4
          ENDIF
          READ(20,9015,END=4000) (UADATE(K),K=1,4),
     2    (UADATA(JUA,K),K=1,15)
          UACNT = UACNT + 1
          IF ( UACNT .EQ. 61 ) UACNT = 1
          GOTO 155
C      if the dates are the same then read the variables for
C      each observed height in the sounding
       ELSEIF ( UACASE .EQ. 0 ) THEN
          JFOUND = II
  180     JUA = JUA + 1
          UAFND = .TRUE.
          IF ( UACNT .EQ. 1 ) THEN
             DO 185 K=1, 3
                READ(20,9001) HEADNG
  185        CONTINUE
             UACNT = 4
          ENDIF
          READ(20,9015,END=4000) (UADATE(K),K=1,4),
     2    (UADATA(JUA,K), K=1,15)
          UACNT = UACNT + 1
          IF ( UACNT .EQ. 61 ) UACNT = 1
```

```
C     compare the dates of the heights where observations
C     were made. If the dates are not the same then the
C     intire profile is stored in UADATA
          CALL DTCOMP(UADATE(1),UADATE(2),UADATE(3),
      2   UADATE(4), YEAR(II),MONTH(II),DAY(II),HOUR(II),
      3   UACASE,0)
          IF ( UACASE .EQ. 0 ) THEN
             GOTO 180
          ELSE
             JUA = JUA + 1
             DO 187 K=1,15
                UADATA(JUA,K) = UADATA(JUA-1,K)
  187        CONTINUE
             DO 190 K=1,6
                UADATA(JUA-1,K) = UADATA(JUA-2,K+6)
  190        CONTINUE
          ENDIF
       ENDIF

 4000 RETURN
 9001 FORMAT(A)
 9015 FORMAT(1X,3(I2,1X),I2,F6.2,F7.0,F7.1,3F7.0,F7.2,F7.0,
     2 F7.1,4F7.0,2F7.2)
C     end of subroutine UATEST
       END
```

7.5.1.4 Subroutine DSTRBN

```
C  SUBROUTINE DSTRBN  ****************************************
C                                                             *
C     Purpose: to count the different combinations of         *
C          precipitation types (rain, snow, and rain with     *
C          snow) for each month of each year. This also       *
C          counts for the distribution of durations, and      *
C          for the distribution of number of hours with       *
C          each time (in hours) from the onset of the         *
C          precipitation event.                               *
C                                                             *
C     Definitions:                                            *
C          EVENTS - see METSTAT                               *
C          HOUR1  - see METSTAT                               *
C          HOUR2  - see METSTAT                               *
C          K      - counter equal to number of hours from     *
C                   the onset of the precipitation event      *
C          LGCRN  - see METSTAT                               *
C          LGCRS  - see METSTAT                               *
C          LGCSNW - see METSTAT                               *
C          MNDIS  - see METSTAT                               *
C          MONTH  - see METSTAT                               *
C          RAIN   - see METSTAT                               *
C          SNOW   - see METSTAT                               *
C          TIMSET - see METSTAT                               *
C          UA     - see METSTAT                               *
```

```
C          YEAR   - see METSTAT                                        *
C          YEAR1  - see METSTAT                                        *
C          YRNDEX - see METSTAT                                        *
C                                                                      *
C     Programmed by Mark Bourassa   1988                               *
C                                                                      *
C**********************************************************************

      SUBROUTINE DSTRBN( TIMSET, UA, LGCRN, LGCSNW, LGCRS,
     2 EVENTS, HOUR1, HOUR2, MNDIS, YEAR, MONTH, YRNDEX,
     2 YEAR1, SNOW, RAIN )

      LOGICAL TIMSET, UA, LGCRN, LGCSNW, LGCRS
      INTEGER EVENTS(62), HOUR1, HOUR2, MNDIS(4,30,12),
     2 YEAR(60), 2 MONTH(60), YRNDEX, YEAR1, K
      REAL SNOW(60), RAIN(60)

C     for each hour add one to the appropriate counter and
C     one to the counter for the monthly total
      DO 200 K=HOUR1, HOUR2, 1
         IF ( ( ( RAIN(K) .GT. 0.0 ) .AND. LGCRN ) .OR.
     2   ( ( SNOW(K) .GT. 0.0 ) .AND. LGCSNW ) .OR.
     3   ( ( RAIN(K) .GT. 0.0 ) .AND. ( SNOW(K) .GT. 0.0 )
     4   .AND. LGCRS ) ) THEN
            IF ( TIMSET .AND. ( .NOT. UA ) ) THEN
               IF ( K .EQ. HOUR1 ) EVENTS(J-1) = EVENTS(J-1)
     5         + 1
            ELSE
               EVENTS(K) = EVENTS(K) + 1
            ENDIF
            IF ( K .EQ. HOUR1 ) EVENTS(61) = EVENTS(61) + 1
            EVENTS(62) = EVENTS(62) + 1

            YRNDEX = YEAR(K) - YEAR1 + 1
            IF ( RAIN(K) .GT. 0.0 ) THEN
               IF ( SNOW(K) .GT. 0.0 ) THEN
                  MNDIS(3,YRNDEX,MONTH(K)) = MNDIS(3,YRNDEX,
     2            MONTH(K)) + 1
               ELSE
                  MNDIS(1,YRNDEX,MONTH(K)) = MNDIS(1,YRNDEX,
     2            MONTH(K)) + 1
               ENDIF
            ELSEIF ( SNOW(K) .GT. 0.0 ) THEN
               MNDIS(2,YRNDEX,MONTH(K)) = MNDIS(2,YRNDEX,
               MONTH(K)) + 1
            ENDIF
            MNDIS(4,YRNDEX,MONTH(K)) = MNDIS(4,YRNDEX,
     2      MONTH(K)) + 1
         ENDIF
  200 CONTINUE

      RETURN
C     end of subroutine DSTRBN
      END
```

7.5.1.5 Subroutine SETVAL

```
C  SUBROUTINE DSTRBN  ****************************************
C                                                              *
C    Purpose: to select the x or y-values:                     *
C         A) Surface Ta [C]                                    *
C         B) Surface Ta - Tw [C]                               *
C         C) Time from onset [hours]                           *
C         D) Duration [hours]                                  *
C         E) Upper air Ta [C]                                  *
C         F) Upper air RH [%]                                  *
C         G) Layer thickness [m]                               *
C         H) Surface Pressure [kPa]                            *
C         I) Surface RH [%]                                    *
C         J) Surface Tw [C]                                    *
C         K) Lapse rate [C/km]                                 *
C         L) Wind Shear [1/C]                                  *
C         M) Ta Difference [C]                                 *
C         N) UA Pressure(sfc) [kPa]                            *
C         O) SFC Wind direction [degrees]                      *
C         P) SFC Wind Speed [m/s]                              *
C         Q) Visibility [km]                                   *
C         R) Snow Precip [cm/hour]                             *
C         Z) Unity.                                            *
C                                                              *
C    Definitions:                                              *
C         DUMMY  - temperary storage location                  *
C         HOUR1  - see METSTAT                                 *
C         HOUR2  - see METSTAT                                 *
C         J      - see METSTAT                                 *
C         JUA    - see METSTAT                                 *
C         K      - counter equal to the number of hours        *
C                  from the onset of the precipitation event   *
C         LGCDDN - see METSTAT                                 *
C         LGCDIR - logical that is true is the direction       *
C                  is being examined                           *
C         LGCTDF - logical that is true if hourly changes      *
C                  are being examined                          *
C         LGCZV  - logical that is true if the other           *
C                  variable is only examined when the this     *
C                  variable is equal to zero.                  *
C         LLAYER - see METSTAT                                 *
C         MAX    - maximum value                               *
C         MIN    - minimum value                               *
C         PRESS  - see METSTAT                                 *
C         RAIN   - see METSTAT                                 *
C         RH     - see METSTAT                                 *
C         SNOW   - see METSTAT                                 *
C         SPEED  - see METSTAT                                 *
C         TA     - see METSTAT                                 *
C         TW     - see METSTAT                                 *
```

```
C         UADATA - see METSTAT                                          *
C         ULAYER - see METSTAT                                          *
C         VAL    - the value being returned                             *
C         VALUE  - see METSTAT                                          *
C         VIS    - see METSTAT                                          *
C         WINDIR - see METSTAT                                          *
C         XYVAR  - logical that is true if variable being               *
C                  set is the x-variable                                *
C                                                                       *
C    Programmed by Mark Bourassa   1988                                 *
C                                                                       *
C***********************************************************************

      SUBROUTINE SETVAL( VALUE, VAL, TA, TW, UADATA, ULAYER,
     2 LLAYER, J, JUA, HOUR1, HOUR2, RH, SPEED, WINDIR,
     3 PRESS, VIS, MIN, MAX, XYVAR, LGCDDN, LGCTDF, LGCZV,
     4 RAIN, SNOW)

      LOGICAL LGCTDF, XYVAR, LGCDDN, LGCZV, LGCDIR
      INTEGER JUA, J, K, HOUR1, HOUR2
      REAL VAL(60), TA(60), TW(60), UADATA(30,15), ULAYER,
     2 LLAYER, RH(60), DUMMY, SNOW(60), RAIN(60), LAYER,
     3 SPEED(60), WINDIR(60), PRESS(60), VIS(60), MIN, MAX
      CHARACTER*1 VALUE

      LGCDIR = .FALSE.
      DO 200 K=HOUR1, HOUR2, 1
         IF ( ( VALUE .EQ. 'A' ) .OR. ( VALUE .EQ. 'a' ) )
     2   THEN
            VAL(K) = TA(K)
         ELSEIF ( ( VALUE .EQ. 'B') .OR. (VALUE .EQ. 'b') )
     2   THEN
            VAL(K) = TA(K) - TW(K)
         ELSEIF ( ( VALUE .EQ. 'C') .OR. (VALUE .EQ. 'c' ) )
     2   THEN
            VAL(K) = K
         ELSEIF ( ( VALUE .EQ. 'D') .OR. (VALUE .EQ. 'd' ) )
     2   THEN
            VAL(K) = J - 1
         ELSEIF ( ( VALUE .EQ. 'E') .OR. (VALUE .EQ. 'e' ) )
     2   THEN
            VAL(K) = UAPT(UADATA,ULAYER,JUA,3)
         ELSEIF ( ( VALUE .EQ. 'F') .OR. (VALUE .EQ. 'f' ) )
     2   THEN
            VAL(K) = UAPT(UADATA,ULAYER,JUA,4)
         ELSEIF ( ( VALUE .EQ. 'G') .OR. (VALUE .EQ. 'g' ) )
     2   THEN
            VAL(K) = LAYER(UADATA,ULAYER,LLAYER,JUA,2)
         ELSEIF ( ( VALUE .EQ. 'H') .OR. (VALUE .EQ. 'h' ) )
     2   THEN
            VAL(K) = PRESS(K)
         ELSEIF ( ( VALUE .EQ. 'I') .OR. (VALUE .EQ. 'i' ) )
     2   THEN
            VAL(K) = RH(K)
```

```
      ELSEIF ( ( VALUE .EQ. 'J') .OR. (VALUE .EQ. 'j' ) )
     2 THEN
         VAL(K) = TW(K)
      ELSEIF ( ( VALUE .EQ. 'K') .OR. (VALUE .EQ. 'k' ) )
     2 THEN
         VAL(K) = LAYER(UADATA,ULAYER,LLAYER,JUA,3)
         DUMMY = LAYER(UADATA,ULAYER,LLAYER,JUA,2)
         IF ( ( DUMMY .NE. -99999.0 ) .AND. ( VAL(K)
     2   .NE. -9999.9 ) ) THEN
            VAL(K) = - 1000.0 * VAL(K) / DUMMY
         ELSE
            VAL(K) = -99999.0
         ENDIF
      ELSEIF ( ( VALUE .EQ. 'L') .OR. (VALUE .EQ. 'l' ) )
     2 THEN
         VAL(K) = LAYER(UADATA,ULAYER,LLAYER,JUA,6)
         DUMMY = LAYER(UADATA,ULAYER,LLAYER,JUA,2)
         IF ( ( DUMMY .NE. -99999.0 ) .AND. ( VAL(K)
     2   .NE. -99999.0 ) ) THEN
            VAL(K) = VAL(K) / DUMMY
         ELSE
            VAL(K) = -99999.0
         ENDIF
      ELSEIF ( ( VALUE .EQ. 'M') .OR. (VALUE .EQ. 'm' ) )
     2 THEN
         VAL(K) = LAYER(UADATA,ULAYER,LLAYER,JUA,3)
      ELSEIF ( ( VALUE .EQ. 'N') .OR. (VALUE .EQ. 'n' ) )
     2 THEN
         VAL(K) = UAPT(UADATA,ULAYER,JUA,1)
      ELSEIF ( ( VALUE .EQ. 'O') .OR. (VALUE .EQ. 'o' ) )
     2 THEN
         VAL(K) = WINDIR(K)
         LGCDIR = .TRUE.
      ELSEIF ( ( VALUE .EQ. 'P') .OR. (VALUE .EQ. 'p' ) )
     2 THEN
         VAL(K) = SPEED(K)
      ELSEIF ( ( VALUE .EQ. 'Q') .OR. (VALUE .EQ. 'q' ) )
     2 THEN
      IF ( ( SNOW(K) .GT. 0.0 ) .AND.
     2    ( RAIN(K) .LE. 0.0 ) ) THEN
            VAL(K) = VIS(K)
      ELSE
         VAL(K) = -99999.0
      ENDIF
      ELSEIF ( ( VALUE .EQ. 'R') .OR. (VALUE .EQ. 'r' ) )
     2 THEN
         IF ( VIS(K) .GT. 0.0 ) THEN
       DUMMY = 0.055 - LOG10( VIS(K) ) / 0.607
       VAL(K) = 10.0 ** DUMMY
         ELSE
            VAL(K) = -99999.0
         ENDIF
      ELSEIF ( ( VALUE .EQ. 'Z') .OR. (VALUE .EQ. 'z' ) )
     2 THEN
```

```
            VAL(K) = 1.0
         ENDIF

C     adjust to code for no data observed
         IF ( VAL(K) .EQ. -9999.9 ) VAL(K) = -99999.0
         IF ( VAL(K) .EQ. -999.99 ) VAL(K) = -99999.0

  200 CONTINUE

C     if the option for hourly differences has been chosen
C     then take the differences
       IF ( LGCTDF ) THEN
       HOUR2 = HOUR2 - 1
         DO 500 K=HOUR1, HOUR2, 1
         IF ( ( VAL(K+1) .NE. -99999.0) .AND.
     2       (VAL(K) .NE. -99999.0 ) ) THEN
               DUMMY = VAL(K+1) - VAL(K)
C     remove the trends
             IF ( ( DUMMY .NE. 0.0 ) .AND.
     2          ( VALUE .EQ. 'A' ) .AND. ( K .LT. 11 ) )
     3          DUMMY = DUMMY + SQRT(0.156 - 0.066 * LOG(K))
             IF ( ( DUMMY .NE. 0.0 ) .AND.
     2          ( VALUE .EQ. 'I' ) .AND. ( I .LT. 6 ) )
     3            DUMMY = DUMMY - 3.172 + 1.955 * LOG(K)
               IF ( LGCDIR .AND. ( DUMMY .LT. -180.0 ) )
     2            DUMMY = DUMMY + 360.0
               IF ( LGCDIR .AND. ( DUMMY .GT. 180.0 ) )
     2            DUMMY = DUMMY - 360.0
               IF ( LGCDDN ) THEN
                  IF ( XYVAR ) THEN
                     WRITE(25,9023) VAL(K), ',', DUMMY
                  ELSE
                     WRITE(26,9023) VAL(K), ',', DUMMY
                  ENDIF
               ENDIF
               VAL(K) = DUMMY
         ELSE
            VAL(K) = -99999.0
         ENDIF
  500    CONTINUE
       HOUR2 = HOUR2 + 1
       ENDIF

C     if the option to examine the other variable only when
C     the value of this variable is zero, then set all non-
C     zero values to the lack of observation code (-9999.9)
       DO 600 K=HOUR1, HOUR2
         IF ( LGCZV .AND. ( VAL(K) .NE. 0.0 ) )
     2    VAL(K) = -99999.0

C     reset MIN and MAX if more extreme values are found
         IF ( ( VAL(K) .LT. MIN ) .AND.
     2    ( VAL(K) .NE. -99999.0 ) ) MIN = VAL(K)
         IF ( ( VAL(K) .GT. MAX ) .AND.
```

```
     2    ( VAL(K) .NE. -99999.0 ) ) MAX = VAL(K)
  600 CONTINUE

      RETURN
 9023 FORMAT( 1X,F9.3,A,3X,F9.3)
C     end of subroutine SETVAL
      END
```

7.5.1.6 Subroutine IDENT

```
C  SUBROUTINE IDENT  ****************************************
C                                                            *
C    Purpose: to select specific x and y-values from each    *
C         precipitation event. The choices are:              *
C              A) all hours of each event                    *
C              B) beginning hour of each event               *
C              C) central hour of each event                 *
C              E) end hour of each event                     *
C              F) first, central, and last hours             *
C              O) only the 'X'-th hour                       *
C              S) skip every 'X' data points.                *
C                                                            *
C    Definitions:                                            *
C         ALLPTS - see METSTAT                               *
C         BSHAVE - see METSTAT                               *
C         ESHAVE - see METSTAT                               *
C         HOUR1  - see METSTAT                               *
C         HOUR2  - see METSTAT                               *
C         J      - see METSTAT                               *
C         JMIN   - see METSTAT                               *
C         K      - counter equal to the number of hours      *
C                  since the onset of the precipitation      *
C                  event                                     *
C         LGCAVE - see METSTAT                               *
C         LGCRN  - see METSTAT                               *
C         LGCRS  - see METSTAT                               *
C         LGCSNW - see METSTAT                               *
C         MINVAL - see METSTAT                               *
C         PTS    - see METSTAT                               *
C         RAIN   - see METSTAT                               *
C         RANGE  - see METSTAT                               *
C         RUNTYP - see METSTAT                               *
C         SKIP   - see METSTAT                               *
C         SNOW   - see METSTAT                               *
C         TIMSET - see METSTAT                               *
C         TMSPAN - see METSTAT                               *
C         UA     - see METSTAT                               *
C         XVAL   - see METSTAT                               *
C         YVAL   - see METSTAT                               *
C                                                            *
C    Programmed by Mark Bourassa 1988                        *
C                                                            *
C************************************************************
```

```
      SUBROUTINE IDENT( TIMSET, UA, RUNTYP, HOUR1, HOUR2,
     2 BSHAVE, ESHAVE, J, SKIP, RAIN, SNOW, XVAL, YVAL,
     3 RANGE, MINVAL, LGCSNW, LGCRN, LGCRS, JMIN, ALLPTS,
     4 PTS, LGCAVE )

      LOGICAL TIMSET, UA, LGCSNW, LGCRN, LGCRS, LGCAVE
      INTEGER HOUR1, HOUR2, BSHAVE, ESHAVE, J, SKIP, JMIN,
     2 TMSPAN, K, ALLPTS(2)
      REAL RAIN(60), SNOW(60), XVAL(60), YVAL(60), RANGE,
     2 MINVAL, PTS(1600,3)
      CHARACTER*1 RUNTYP

      IF (( RUNTYP .EQ. 'A' ) .OR. ( RUNTYP .EQ. 'a' ) .OR.
     2 ( RUNTYP .EQ. 'S' ) .OR. ( RUNTYP .EQ. 's'  ) ) THEN
         DO 400 K=HOUR1,HOUR2,SKIP
            IF ( TIMSET .AND. ( .NOT. UA ) ) THEN
               TMSPAN = J - 1
            ELSE
               TMSPAN = K
            ENDIF
            IF ( (RAIN(K) .GT. 0.0) .AND. (SNOW(K) .GT. 0.0)
     2      .AND. ( XVAL(K) .NE. -99999.0 ) .AND.
     3      ( YVAL(K) .NE. -99999.0 ) ) THEN
               WRITE(10,9005) XVAL(K), ',', YVAL(K), ' ', K
               IF ( LGCRS .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
            ELSEIF ( ( RAIN(K) .GT. 0.0 ) .AND.
     2      ( XVAL(K) .NE. -99999.0 ) .AND.
     3      ( YVAL(K) .NE. -99999.0 ) ) THEN
               WRITE(8,9005) XVAL(K), ',', YVAL(K), ' ', K
               IF ( LGCRN .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
            ELSEIF ( ( XVAL(K) .NE. -99999.0 ) .AND.
     2      ( YVAL(K) .NE. -99999.0 ) ) THEN
               WRITE(9,9005) XVAL(K), ',', YVAL(K), ' ', K
               IF ( LGCSNW .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
            ENDIF
  400    CONTINUE
      ELSEIF ( ( RUNTYP .NE. 'F') .AND. (RUNTYP .NE. 'f' ) )
     2    THEN
         IF ( ( RUNTYP .EQ. 'B' ) .OR. ( RUNTYP .EQ. 'b' ) )
     2    THEN
            K = BSHAVE
         ELSEIF ((RUNTYP .EQ. 'C') .OR. (RUNTYP .EQ. 'c' ) )
     2    THEN
            K = MOD(J+1,2)
         ELSEIF ((RUNTYP .EQ. 'E') .OR. (RUNTYP .EQ. 'e' ) )
     2    THEN
            K = J - ESHAVE
         ELSE
            K = JMIN
         ENDIF
```

```
      IF ( TIMSET .AND. ( .NOT. UA ) ) THEN
         TMSPAN = J - 1
      ELSE
         TMSPAN = K
      ENDIF
      IF ( ( K .GE. HOUR1 ) .AND. ( K .LE. HOUR2 ) ) THEN
         IF ( (RAIN(K) .GT. 0.0) .AND. (SNOW(K) .GT. 0.0)
     2      .AND. ( XVAL(K) .NE. -99999.0 ) .AND.
     3      ( YVAL(K) .NE. -99999.0 ) ) THEN
            WRITE(10,9005) XVAL(K), ',', YVAL(K)
            IF ( LGCRS .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
         ELSEIF ( ( RAIN(K) .GT. 0.0 ) .AND.
     2      ( XVAL(K) .NE. -99999.0 ) .AND.
     3      ( YVAL(K) .NE. -99999.0 ) ) THEN
            WRITE(8,9005) XVAL(K), ',', YVAL(K)
            IF ( LGCRN .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
         ELSEIF ( ( XVAL(K) .NE. -99999.0 ) .AND.
     2      ( YVAL(K) .NE. -99999.0 ) ) THEN
            WRITE(9,9005) XVAL(K), ',', YVAL(K)
            IF ( LGCSNW .AND. LGCAVE ) CALL
     2         PREAVE( ALLPTS, PTS, XVAL(K), YVAL(K), K )
         ENDIF
      ENDIF
   ELSE

      K = MOD(J+1,2)
      IF ( ( RAIN(BSHAVE) .GT. 0.0) .AND. (SNOW(BSHAVE)
     2   .GT. 0.0) )  THEN
         WRITE(10,9005) XVAL(BSHAVE), ',', YVAL(BSHAVE)
      ELSEIF ( RAIN(SHAVE) .GT. 0.0 ) THEN
         WRITE(8,9005) XVAL(BSHAVE), ',', YVAL(BSHAVE)
      ELSE
         WRITE(9,9005) XVAL(BSHAVE), ',', YVAL(BSHAVE)
      ENDIF
      IF ( ( RAIN(K) .GT. 0.0) .AND. (SNOW(K) .GT. 0.0) )
     2   THEN
         WRITE(13,9005) XVAL(K), ',', YVAL(K)
      ELSEIF ( RAIN(K) .GT. 0.0 ) THEN
         WRITE(11,9005) XVAL(K), ',', YVAL(K)
      ELSE
         WRITE(12,9005) XVAL(K), ',', YVAL(K)
      ENDIF
      IF ( ( RAIN(SHAVE) .GT. 0.0) .AND. (SNOW(SHAVE)
     2   .GT. 0.0 ) ) THEN
         WRITE(16,9005) XVAL(ESHAVE), ',', YVAL(ESHAVE)
      ELSEIF ( RAIN(SHAVE) .GT. 0.0 ) THEN
         WRITE(14,9005) XVAL(ESHAVE), ',', YVAL(ESHAVE)
      ELSE
         WRITE(15,9005) XVAL(ESHAVE), ',', YVAL(ESHAVE)
      ENDIF
   ENDIF
   RETURN
```

```
 9005 FORMAT(1X,F8.2,A,F8.2,A,I2)
C     end of subroutine IDENT
      END
```

7.5.1.7 Subroutine SPEAK

```
C  SUBROUTINE SPEAK.FOR  ************************************
C                                                             *
C    Purpose: to interactively set the analysis options.      *
C                                                             *
C    Definitions:                                             *
C         ALTHR1 - see METSTAT                                *
C         ALTHR2 - see METSTAT                                *
C         ALTT1  - see METSTAT                                *
C         ALTT2  - see METSTAT                                *
C         BHOUR  - see METSTAT                                *
C         BSHAVE - see METSTAT                                *
C         DPTS   - see METSTAT                                *
C         EHOUR  - see METSTAT                                *
C         ESHAVE - see METSTAT                                *
C         IDUMMY - an integer temperary storage location      *
C         JMAX   - see METSTAT                                *
C         JMIN   - see METSTAT                                *
C         LBSHAV - logical that is true if the option has     *
C                  been chosen to ignore the first hour of    *
C                  each precipitation event                   *
C         LESHAV - logical that is true if the option has     *
C                  been chosen to ignore the last hour of     *
C                  each precipitation event                   *
C         LGCAVE - see METSTAT                                *
C         LGCDDN - see METSTAT                                *
C         LGCHIS - see METSTAT                                *
C         LGCMIN - see METSTAT                                *
C         LGCNRM - see METSTAT                                *
C         LGCPCP - see METSTAT                                *
C         LGCPRO - see METSTAT                                *
C         LGCRN  - see METSTAT                                *
C         LGCRS  - see METSTAT                                *
C         LGCSNW - see METSTAT                                *
C         LGCWET - see METSTAT                                *
C         LGCXTD - see METSTAT                                *
C         LGCYTD - see METSTAT                                *
C         LGCZRO - see METSTAT                                *
C         LGCZXV - see METSTAT                                *
C         LGCZYV - see METSTAT                                *
C         LLAYER - see METSTAT                                *
C         MINVAL - see METSTAT                                *
C         MODE   - a character variable used when asking      *
C                  whether the data should be sorted by the   *
C                  duration of the event, by the number of    *
C                  hours from the onset of the event, or by   *
C                  both                                       *
C         OHOUR  - if only one hour of each event, defined    *
```

```
C                by the number of hours from onset, is of  *
C                interest, then this is the number of      *
C                hours from onset                          *
C          RANGE  - see METSTAT                            *
C          RESPON - character response to yes/no queries   *
C          RUNTYP - see METSTAT                            *
C          SKIP   - see METSTAT                            *
C          START  - see METSTAT                            *
C          TIMSET - see METSTAT                            *
C          UA     - see METSTAT                            *
C          ULAYER - see METSTAT                            *
C          WIDTH  - see METSTAT                            *
C          X      - character description of the           *
C                (horizontal) x-variable                   *
C          XVALUE - see METSTAT                            *
C          Y      - character description of the (vertical) *
C                y-variable                                *
C          YVALUE - see METSTAT                            *
C                                                          *
C     Programmed by Mark Bourassa   1988                   *
C                                                          *
C**********************************************************

      SUBROUTINE SPEAK(LGCRN,LGCSNW,LGCRS,LGCMIN,LGCAVE,
     1 LGCPCP,START,JMIN,JMAX,BSHAVE,ESHAVE,SKIP,DPTS,WIDTH,
     2 X,Y,XVALUE,YVALUE,RUNTYP,S,RANGE,MINVAL,TIMSET,BHOUR,
     3 EHOUR,ULAYER,LLAYER,UA,ALTT1,ALTT2,ALTHR1,ALTHR2,
     4 LGCZRO,LGCHIS,LGCNRM,LGCDDN,LGCXTD,LGCYTD,LGCPRO,
     5 LGCWET,LGCZXV,LGCZYV)

      LOGICAL LGCRN, LGCSNW, LGCRS, LGCMIN, LGCAVE, LGCPCP,
     + LBSHAV, LESHAV, TIMSET, UA, LGCZRO, LGCHIS, LGCDDN,
     + LGCNRM, LGCXTD, LGCYTD, LGCPRO, LGCWET, LGCZXV,
     + LGCZYV
      INTEGER I, J, START, JMIN, JMAX, BSHAVE, ESHAVE, SKIP,
     + DPTS, OHOUR, EHOUR, BHOUR, IDUMMY, ALTT1, ALTT2,
     + ALTHR1, ALTHR2
      REAL WIDTH, RANGE, MINVAL, ULAYER, LLAYER
      CHARACTER X*7, Y*2, MODE*15
      CHARACTER*1 XVALUE, YVALUE, RESPON, RUNTYP

      OPEN(UNIT = 4, FILE = 'RHVTA.BAT', STATUS = 'UNKNOWN')

      SKIP = 1
      START = 0
      LGCMIN = .FALSE.
      LGCAVE = .FALSE.
      LGCPCP = .FALSE.
      LGCRN = .TRUE.
      LGCSNW = .TRUE.
      LGCRS = .TRUE.
      LGCZRO = .FALSE.
      LGCHIS = .FALSE.
```

```
      LGCNRM = .TRUE.
      LGCDDN = .FALSE.
      LGCPRO = .FALSE.
      LGCWET = .FALSE.
      UA = .FALSE.
      RANGE = 8.0
      MINVAL = -2.0
      TIMSET = .TRUE.

      WRITE(6,9003) 'CHOOSE A TIME (HOURS XX) WHEN AN HOUR',
     2 ' OR MORE'
      WRITE(6,9003) 'CAN PASS WITHOUT PRECIPITATION WITHIN',
     2 ' AN EVENT'
      READ(4,9004) ALTT1
      WRITE(6,9003) 'HOW MANY HOURS CAN PASS (X)?'
      READ(4,9022) ALTHR1
      ALTHR1 = ALTHR1 + 1
      WRITE(6,9003) 'CHOOSE ANOTHER TIME (HOURS XX) WHEN',
     2 ' AN HOURS'
      WRITE(6,9003) 'CAN PASS WITHOUT PRECIPITATION WITHIN',
     2 ' AN EVENT'
      READ(4,9004) ALTT2
      WRITE(6,9003) 'HOW MANY HOURS CAN PASS (X)?'
      READ(4,9022) ALTHR2
      ALTHR2 = ALTHR2 + 1

   20 WRITE(6,9003) 'CHOOSE CATAGORIZATION BY:  A) DURATION'
      WRITE(6,9003) '                           B) TIME ',
     2 'FROM ONSET'
      WRITE(6,9003) '                           C) SET BOTH'
      READ(4,9001) RESPON
      IF (( RESPON .EQ. 'A' ) .OR. ( RESPON .EQ. 'C' )) THEN
         TIMSET = .TRUE.
      ELSEIF ( RESPON .EQ. 'B' ) THEN
         TIMSET = .FALSE.
      ELSE
         WRITE(6,9003) 'TRY AGAIN'
         GOTO 20
      ENDIF

   50 IF ( TIMSET ) THEN
         MODE = '   DURATION   '
      ELSE
         MODE = 'HOUR FROM ONSET'
      ENDIF
      WRITE(6,9003) 'ENTER MINIMUM ', MODE, ' OF THE EVENT'
      IF ( TIMSET ) THEN
         READ(4,9004) JMIN
      ELSE
         READ(4,9004) BHOUR
         IF ( RESPON .EQ. 'B' ) JMIN = BHOUR
      ENDIF

      WRITE(6,9003) 'ENTER MAXIMUM ', MODE, ' OF THE EVENT'
```

```
      IF ( TIMSET ) THEN
         READ(4,9004) JMAX
         IF ( JMAX .LT. JMIN ) THEN
            WRITE(6,9003) 'TRY AGAIN'
            GOTO 50
         ENDIF
      ELSE
         READ(4,9004) EHOUR
         IF ( EHOUR .LT. BHOUR ) THEN
            WRITE(6,9003) 'TRY AGAIN'
            GOTO 50
         ENDIF
         IF ( RESPON .EQ. 'B' ) JMAX = 60
      ENDIF

      IF ( JMAX .GT. 60 ) JMAX = 60

      IF ( RESPON .EQ. 'C' ) THEN
         TIMSET = .FALSE.
         RESPON = 'D'
         GOTO 50
      ENDIF

   60 WRITE(6,9003) 'The ''Y'' variable will be:  A)  ',
     2 'Raw data'
      WRITE(6,9003) '                                B)  ',
     2 'Time Differences'
      READ(4,9001) RESPON
      IF ( RESPON .EQ. 'A' ) THEN
         LGCYTD = .FALSE.
      ELSEIF ( RESPON .EQ. 'B' ) THEN
         LGCYTD = .TRUE.
      ELSE
         GOTO 60
      ENDIF
   65 WRITE(6,9003) 'The ''Y'' variable will be:  A)  ',
     2 'Unconditional'
      WRITE(6,9003) '                                B)  ''Y'' ',
     2 '= 0 only'
      READ(4,9001) RESPON
      IF (( RESPON .EQ. 'A' ) .OR. ( RESPON .EQ. 'a' )) THEN
      LGCZYV = .FALSE.
      ELSEIF (( RESPON .EQ. 'B' ) .OR. ( RESPON .EQ. 'b' ))
     2 THEN
         LGCZYV = .TRUE.
      ELSE
         GOTO 65
      ENDIF
      CALL VARABS(YVALUE, Y, MINVAL, RANGE, LLAYER, ULAYER,
     2 'Y', UA)

   70 WRITE(6,9003) 'The ''X'' variable will be:  A)  Raw ',
     2 'data'
      WRITE(6,9003) '                                B)  Time ',
```

```
     2 'Differences'
      READ(4,9001) RESPON
      IF ( RESPON .EQ. 'A' ) THEN
         LGCXTD = .FALSE.
      ELSEIF ( RESPON .EQ. 'B' ) THEN
         LGCXTD = .TRUE.
      ELSE
         GOTO 70
      ENDIF

     75 WRITE(6,9003) 'The ''X'' variable will be:  A)  ',
       2 'Unconditional'
      WRITE(6,9003) '                                B)  ''X'' ',
       2 '= 0 only'
      READ(4,9001) RESPON
      IF (( RESPON .EQ. 'A' ) .OR. ( RESPON .EQ. 'a' )) THEN
      LGCZXV = .FALSE.
      ELSEIF ((RESPON .EQ. 'B') .OR. (RESPON .EQ. 'b')) THEN
         LGCZXV = .TRUE.
      ELSE
         GOTO 75
      ENDIF
      CALL VARABS(XVALUE, X, MINVAL, RANGE, LLAYER, ULAYER,
       2 'X', UA)

    500 WRITE(6,9003) 'LIST A) ALL HOURS OF EACH EVENT'
      WRITE(6,9003) '     B) BEGINNING HOUR OF EACH EVENT'
      WRITE(6,9003) '     C) CENTRAL HOUR OF EACH EVENT'
      WRITE(6,9003) '     E) END HOUR OF EACH EVENT'
      WRITE(6,9003) '     F) FIRST, CENTRAL, AND LAST HOURS'
      WRITE(6,9003) '     O) ONLY THE ''X''-TH HOUR'
      WRITE(6,9003) '     S) SKIP EVERY ''X'' DATA POINTS'
      READ(4,9001) RUNTYP
      IF ( ( RUNTYP .EQ. 'A' ) .OR. ( RUNTYP .EQ. 'a' ) .OR.
     2     ( RUNTYP .EQ. 'B' ) .OR. ( RUNTYP .EQ. 'b' ) .OR.
     3     ( RUNTYP .EQ. 'C' ) .OR. ( RUNTYP .EQ. 'c' ) .OR.
     4     ( RUNTYP .EQ. 'E' ) .OR. ( RUNTYP .EQ. 'e' ) .OR.
     5     ( RUNTYP .EQ. 'O' ) .OR. ( RUNTYP .EQ. 'o' ) .OR.
     6     ( RUNTYP .EQ. 'S' ) .OR. ( RUNTYP .EQ. 's')) THEN
         WRITE(8,9003) ' "Rain" '
         WRITE(9,9003) ' "Snow" '
         WRITE(10,9003) ' "Rain and Snow" '
         IF ((RUNTYP .EQ. 'A') .OR. (RUNTYP .EQ. 'a')) THEN
            IF ( TIMSET ) THEN
               WRITE(17,9003) ' TITLE "',Y,' vs. ',X,
     2          ': All Hours",'
            ELSEIF ( BHOUR .NE. EHOUR ) THEN
               WRITE(17,9018) ' TITLE "',Y,' VS. ',X,
     2          ': Hours ',BHOUR,' to ',EHOUR,'",'
            ELSE
               WRITE(17,9018) ' TITLE "',Y,' VS. ',X,
     2          ': Hour ',BHOUR,'",'
            ENDIF
         ELSEIF ( ( RUNTYP .EQ. 'B' ) .OR.
```

```
     2    ( RUNTYP .EQ. 'b' ) ) THEN
             WRITE(17,9003) ' TITLE "',Y,' vs. ',X,
     2       ': Beginning Hours,'
          ELSEIF ( ( RUNTYP .EQ. 'C' ) .OR.
     2    ( RUNTYP .EQ. 'c' ) ) THEN
             WRITE(17,9003) ' TITLE "',Y,' vs. ',X,
     2       ': Central Hours",'
          ELSEIF ( ( RUNTYP .EQ. 'E' ) .OR.
     2    ( RUNTYP .EQ. 'e' ) ) THEN
             WRITE(17,9003) ' TITLE "',Y,' vs. ',X,
     2       ': Ending Hours",'
          ELSEIF ( ( RUNTYP .EQ. 'O' ) .OR.
     2    ( RUNTYP .EQ. 'o' ) ) THEN
   510       WRITE(6,9003) 'CHOOSE ''X'':'
             READ(4,9004) OHOUR
             IF ( ( OHOUR .LT. JMIN ) .OR.
     2       ( OHOUR .GT. JMAX ) ) GOTO 510
             JMIN=OHOUR
             WRITE(17,9018) ' TITLE "',Y,' vs. ',X,': Hour ',
     2       JMIN,'",'
          ELSE
             WRITE(6,9003) ' USING EVERY X-TH DATA POINT. ',
     2       'ENTER X: '
             READ(4,9004) SKIP
   520       WRITE(6,9003) 'START WITH WHICH DATA POINT IN ',
             'AN EVENT?'
             WRITE(6,9006) 'CHOOSE 1 THROUGH ', SKIP,':'
             READ(4,9004) START
             IF ( START .GT. SKIP ) GOTO 520
             WRITE(17,9003) ' TITLE "      ', Y,' vs. ', X
     2       , '",'
          ENDIF
        ELSEIF ( ( RUNTYP .EQ. 'F' ) .OR.
     2  ( RUNTYP .EQ. 'f' ) ) THEN
          WRITE(8,9003) ' "Rain: beginning of event" '
          WRITE(9,9003) ' "Snow: beginning of event" '
          WRITE(10,9003) ' "Rain and Snow: beginning of ',
     2    'event" '
          WRITE(11,9003) ' "Rain: middle of event" '
          WRITE(12,9003) ' "Snow: middle of event" '
          WRITE(13,9003) ' "Rain and Snow: middle of event" '
          WRITE(14,9003) ' "Rain: end of event" '
          WRITE(15,9003) ' "Snow: end of event" '
          WRITE(16,9003) ' "Rain and Snow: end of event" '
          WRITE(17,9003) ' TITLE "      ', Y,' vs. ', X, '",'
        ELSE
          WRITE(6,9003) ' TRY AGAIN! '
          GOTO 500
        ENDIF
        IF ( JMIN .EQ. JMAX ) THEN
          WRITE(17,9006) ' "   Duration: ', JMIN, ' hours",'
        ELSE
          WRITE(17,9007) ' "   Duration: ',JMIN,' to ',JMAX,
     2    'hours",'
```

```
      ENDIF

  600 WRITE(6,9003) 'USE: A) EVENTS WITH UPPER AIR PROFILES'
      WRITE(6,9003) '     B) ALL EVENTS'
      READ(4,9001) RESPON
      IF (( RESPON .EQ. 'A' ) .OR. ( RESPON .EQ. 'a' )) THEN
         LGCPRO = .TRUE.
         UA = .FALSE.
      ELSEIF ( ( RESPON .NE. 'B' ) .AND.
     2 ( RESPON .NE. 'b' ) ) THEN
         WRITE(6,9003) 'TRY AGAIN!'
         GOTO 600
      ENDIF

      LBSHAV = .FALSE.
      LESHAV = .TRUE.
  700 IF ( LBSHAV ) THEN
         WRITE(6,9003) 'BOUNDRY OPTIONS:  A) ',
     2 'BEGINNING HOURS   REMOVED'
      ELSE
         WRITE(6,9003) 'BOUNDRY OPTIONS:  A) ',
     2 'BEGINNING HOURS   INCLUDED'
      ENDIF
      IF ( LESHAV ) THEN
         WRITE(6,9003) '                  B) ENDING ',
     2 'HOURS       REMOVED'
      ELSE
         WRITE(6,9003) '                  B) ENDING ',
     2 'HOURS       INCLUDED'
      ENDIF
      WRITE(6,9003) '                C) CONTINUE AS IS'
      READ(4,9001) RESPON
      IF (( RESPON .EQ. 'A' ) .OR. ( RESPON .EQ. 'a' )) THEN
         LBSHAV = .NOT. LBSHAV
         GOTO 700
      ELSEIF ( ( RESPON .EQ. 'B' ) .OR.
     2 ( RESPON .EQ. 'b' ) ) THEN
         LESHAV = .NOT. LESHAV
         GOTO 700
      ELSEIF ( ( RESPON .NE. 'C' ) .AND.
     2 ( RESPON .NE. 'c' ) ) THEN
         WRITE(6,9003) 'TRY AGAIN!'
         GOTO 700
      ENDIF
      IF ( LBSHAV ) THEN
         BSHAVE = 2
      ELSE
         BSHAVE = 1
      ENDIF
      IF ( LESHAV ) THEN
         ESHAVE = 2
      ELSE
         ESHAVE = 1
      ENDIF
```

```
      IF ( LBSHAV .AND. LESHAV ) THEN
         WRITE(17,9003) ' "((Bounding hours removed))" '
      ELSEIF ( LBSHAV ) THEN
         WRITE(17,9003) ' "((BEGINNING HOURS REMOVED))" '
      ELSE
         WRITE(17,9003) ' "((ENDING HOURS REMOVED))" '
      ENDIF
      WRITE(17,9003) '.'

      IF (( RUNTYP .NE. 'F') .AND. ( RUNTYP .NE. 'f' )) THEN
         WRITE(6,9003) 'ENNABLE STATISTICAL MINIMUM ',
     2   'OPTION (Y/N)?'
         READ(4,9001) RESPON
         IF ( ( RESPON .EQ. 'Y' ) .OR.
     2   ( RESPON .EQ. 'y' ) ) THEN
            LGCMIN = .TRUE.
            LGCSNW = .FALSE.
            WRITE(6,9003) 'ENTER WIDTH OF BLOCKS
     2      (in X.XX degrees C)'
2000        READ(4,9009) WIDTH
            DPTS = INT( RANGE / WIDTH + 0.5 )
            IF ( DPTS .GT. 100 ) THEN
               WRITE(6,9003) 'WIDTH TOO SMALL, TRY AGAIN'
               GOTO 2000
            ENDIF

2025        WRITE(6,9003) 'DO YOU WANT:  A) MAXIMAL POINTS'
            WRITE(6,9003) '              B) MINIMAL POINTS'
            READ(4,9001) RESPON
            IF ( ( RESPON .EQ. 'A' ) .OR.
     2      ( RESPON .EQ. 'a' ) ) THEN
               S = -1.0
               WRITE(19,9003) ' "Maxima" '
            ELSEIF ( ( RESPON .EQ. 'B' ) .OR.
     2      ( RESPON .EQ. 'b' )) THEN
               S = 1.0
               WRITE(19,9003) ' "Minima" '
            ELSE
               WRITE(6,9005) 'TRY AGAIN'
               GOTO 2025
            ENDIF
         ENDIF

3000     WRITE(6,9003) 'ENNABLE STATISTICAL MEAN OPTION ',
     2   '(Y/N)?'
         READ(4,9001) RESPON
         IF ( ( RESPON .EQ. 'Y' ) .OR.
     2   ( RESPON .EQ. 'y' ) ) THEN
            LGCAVE = .TRUE.
4100        WRITE(6,9003) 'Ignore differences of zero ',
     2      '(Y/N)?'
            READ(4,9001) RESPON
            IF (( RESPON .EQ. 'Y' ) .OR.
     2      ( RESPON .EQ. 'y' ) ) THEN
```

```
            LGCZRO = .TRUE.
          ELSEIF ((RESPON .NE. 'N') .AND.
     2    (RESPON .NE. 'n')) THEN
            WRITE(6,9003) 'TRY AGAIN (Y/N)'
            GOTO 4100
          ENDIF
4200      WRITE(6,9003) 'Produce a histogam (Y/N)?'
          READ(4,9001) RESPON
          IF (( RESPON .EQ. 'Y' ) .OR.
     2    ( RESPON .EQ. 'y' ) ) THEN
            LGCHIS = .TRUE.
4300        WRITE(6,9003) 'Normalize the histogram ',
     2      '(Y/N)?'
            READ(4,9001) RESPON
            IF (( RESPON .EQ. 'Y' ) .OR.
     2      ( RESPON .EQ. 'y' ) ) THEN
              LGCNRM = .TRUE.
            ELSEIF ((RESPON .NE. 'N') .AND.
     2      (RESPON .NE. 'n')) THEN
              WRITE(6,9003) 'TRY AGAIN (Y/N)'
              GOTO 4300
            ENDIF
          ELSEIF ((RESPON .NE. 'N') .AND.
     2    (RESPON .NE. 'n')) THEN
            WRITE(6,9003) 'TRY AGAIN (Y/N)'
            GOTO 4200
          ENDIF
4400      WRITE(6,9003) 'Output difference vs. raw ',
     2    'value (Y/N)?'
          READ(4,9001) RESPON
          IF (( RESPON .EQ. 'Y' ) .OR.
     2    ( RESPON .EQ. 'y' ) ) THEN
            LGCDDN = .TRUE.
          ELSEIF ((RESPON .NE. 'N') .AND.
     2    (RESPON .NE. 'n')) THEN
            WRITE(6,9003) 'TRY AGAIN (Y/N)'
            GOTO 4400
          ENDIF
        ELSEIF ( ( RESPON .NE. 'N' ) .AND.
     2  ( RESPON .NE. 'n' ) ) THEN
          WRITE(6,9003) 'TRY AGAIN! (Y/N)'
          GOTO 3000
        ENDIF


        IF ( LGCMIN .OR. LGCAVE ) THEN
5000      IF ( LGCRN ) THEN
            WRITE(6,9003) 'WITH RESPSCT TO: A) RAIN ',
     2      '              TRUE'
          ELSE
            WRITE(6,9003) 'WITH RESPECT TO: A) RAIN  ',
     2      '             FALSE'
          ENDIF
          IF ( LGCSNW ) THEN
```

```
            WRITE(6,9003) '                    B) SNOW ',
     2        '            TRUE'
            ELSE
              WRITE(6,9003) '                  B) SNOW  ',
     2        '            FALSE'
            ENDIF
            IF ( LGCRS ) THEN
               WRITE(6,9003) '                  C) RAIN ',
     2        'WITH SNOW   TRUE'
            ELSE
              WRITE(6,9003) '                  C) RAIN ',
     2        'WITH SNOW   FALSE'
            ENDIF
            IF ( LGCWET ) THEN
               WRITE(6,9003) '                  D) WET SNOW',
     2        ' ONLY    TRUE'
            ELSE
              WRITE(6,9003) '                  D) WET SNOW ',
     2        'ONLY    FALSE'
            ENDIF
            IF ( LGCRN .OR. LGCSNW .OR. LGCRS ) THEN
               WRITE(6,9003) '                   E) CONTINUE',
     2        ' AS IS'
            ENDIF
            READ(4,9001) RESPON
            IF ( ( RESPON .EQ. 'A' ) .OR.
     2           ( RESPON .EQ. 'a' ) ) THEN
               LGCRN = .NOT. LGCRN
               GOTO 5000
            ELSEIF ( ( RESPON .EQ. 'B' ) .OR.
     2           ( RESPON .EQ. 'b' ) ) THEN
               LGCSNW = .NOT. LGCSNW
               GOTO 5000
            ELSEIF ( ( RESPON .EQ. 'C' ) .OR.
     2           ( RESPON .EQ. 'c' ) ) THEN
               LGCRS = .NOT. LGCRS
               GOTO 5000
            ELSEIF ( ( RESPON .EQ. 'D' ) .OR.
     2           ( RESPON .EQ. 'd' ) ) THEN
               LGCWET = .NOT. LGCWET
               GOTO 5000
            ELSEIF ( ( LGCRN .OR. LGCSNW .OR. LGCRS ) .AND.
     2           ( ( RESPON .NE. 'E' ) .AND.
     2           ( RESPON .NE. 'e' ) ) ) THEN
               WRITE(6,9003) 'TRY AGAIN!'
               GOTO 5000
            ENDIF
         ENDIF
      ENDIF

 6000 WRITE(6,9003) 'ENNABLE MONTHLY PRECIPITATION TOTALS ',
     2 '(Y/N)?'
      READ(4,9001) RESPON
      IF ( ( RESPON .EQ. 'Y' ) .OR.
```

```
     2 ( RESPON .EQ. 'y' ) ) THEN
         LGCPCP = .TRUE.
       ELSEIF ( ( RESPON .NE. 'N' ) .AND.
     2 ( RESPON .NE. 'n' ) ) THEN
         WRITE(6,9003) 'TRY AGAIN! (Y/N)'
         GOTO 6000
       ENDIF

 9001 FORMAT(A)
 9002 FORMAT(2X,3(I2,3X),1X,I2,2X,F8.1,10X,3(2X,F8.1),12X,
     2 F8.1)
 9003 FORMAT(1X,A)
 9004 FORMAT(I2)
 9005 FORMAT(1X,F8.2,A,F8.2)
 9006 FORMAT(1X,A,I2,A)
 9007 FORMAT(1X,A,I2,A,I2,1X,A)
 9003 FORMAT(1X,A,A,A,A,A)
 9009 FORMAT(F4.2)
 9018 FORMAT(1X,A,A,A,A,A,I2,A,I2,A)
 9021 FORMAT(F5.0)
 9022 FORMAT(I1)

 9999 RETURN
C     end of subroutine SPEAK
       END
```

7.5.1.8 Subroutine VARABS

```
C  SUBROUTINE VARABS  ****************************************
C                                                             *
C     Purpose: to interactively determine the variable for    *
C          an axis.                                           *
C                                                             *
C     Definitions:                                            *
C          LABEL  - character label for the variable.         *
C                 Passed to X and Y in subroutine SPEAK       *
C          LLAYER - see METSTAT                               *
C          MINVAL - see METSTAT                               *
C          RANGE  - see METSTAT                               *
C          UA     - see METSTAT                               *
C          ULAYER - see METSTAT                               *
C          VALUE  - character response to the selection of    *
C                 the variable. Passed to XVALUE or YVALUE    *
C          XY     - character 'X' or 'Y' depending on whcih   *
C                 variable is being selected                  *
C                                                             *
C     Programmed by Mark Bourassa  1988                       *
C                                                             *
C*************************************************************

       SUBROUTINE VARABS( VALUE, LABEL, MINVAL, RANGE,
     2 LLAYER, ULAYER, XY, UA )
```

```
      LOGICAL UA
      REAL MINVAL, RANGE, LLAYER, ULAYER
      CHARACTER VALUE*1, LABEL*10, XY*1

   55 WRITE(6,9003) 'CHOOSE ',XY,' AXIS:   A) Surface Ta'
      WRITE(6,9003) '                      B) Surface Ta - Tw'
      WRITE(6,9003) '                      C) Time from onset'
      WRITE(6,9003) '                      D) Duration'
      WRITE(6,9003) '                      E) Upper air Ta'
      WRITE(6,9003) '                      F) Upper air RH'
      WRITE(6,9003) '                      G) Layer thickness'
      WRITE(6,9003) '                      H) Surface Pressure'
      WRITE(6,9003) '                      I) Surface RH'
      WRITE(6,9003) '                      J) Surface Tw'
      WRITE(6,9003) '                      K) Lapse rate'
      WRITE(6,9003) '                      L) Wind Shear'
      WRITE(6,9003) '                      M) Ta Difference'
      WRITE(6,9003) '                      N) UA Pressure(sfc)'
      WRITE(6,9003) '                      O) SFC Wind direction'
      WRITE(6,9003) '                      P) SFC Wind Speed'
      WRITE(6,9003) '                      Q) Visibility'
      WRITE(6,9003) '                      R) Snow Precip'
      WRITE(6,9003) '                      Z) Unity'
      READ(4,9001) VALUE
      IF ( ( VALUE .EQ. 'A' ) .OR. ( VALUE .EQ. 'a' ) ) THEN
         LABEL = 'Ta'
         WRITE(17,9003) XY,' LABEL "Air Temperature ((C))".'
         WRITE(17,9003) XY,' MIN -2.0, MAX 6.0.'
      ELSEIF ((VALUE .EQ. 'B' ) .OR. ( VALUE .EQ. 'b')) THEN
         LABEL = 'Ta - Tw'
         WRITE(17,9003) XY,' LABEL "Ta - Tw ((C))".'
         WRITE(17,9003) 'X MIN -0.0, MAX 8.0.'
         MINVAL = 0.0
      ELSEIF ((VALUE .EQ. 'C' ) .OR. ( VALUE .EQ. 'c')) THEN
         LABEL = 'Time'
         RANGE = 30.0
         MINVAL = 0.0
         WRITE(17,9003) XY,' LABEL "Time From Onset ',
     2   '((hours))".'
         WRITE(17,9003) XY,' MIN 0, MAX 25.0.'
      ELSEIF ((VALUE .EQ. 'D' ) .OR. ( VALUE .EQ. 'd')) THEN
         LABEL = 'Duration'
         RANGE = 30.0
         MINVAL = 0.0
         WRITE(17,9003) XY,' LABEL "Duration ((hours))".'
         WRITE(17,9003) XY,' MIN 0, MAX 25.0.'
      ELSEIF ((VALUE .EQ. 'E' ) .OR. ( VALUE .EQ. 'e')) THEN
         UA = .TRUE.
         RANGE = 35.0
         MINVAL = -25.0
         WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2   'LAYER (XXXX.):'
         READ(4,9021) ULAYER
         LABEL = 'Ta(P)'
```

```
      IDUMMY = INT( ULAYER )
      WRITE(17,9006) XY,' LABEL "Ta(P=', IDUMMY, 'mb) ',
     2  '((C))".'
      WRITE(17,9003) XY,' MIN -20.0, MAX 10.0.'
     ELSEIF ((VALUE .EQ. 'F' ) .OR. ( VALUE .EQ. 'F')) THEN
      UA = .TRUE.
      RANGE = 100.0
      MINVAL = 0.0
      WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2  'LAYER (XXXX.):'
      READ(4,9021) ULAYER
      LABEL = 'RH(P)'
      IDUMMY = INT( ULAYER )
      WRITE(17,9006) XY,' LABEL "RH(P=', IDUMMY, 'mb) ',
     2  '((C))".'
      WRITE(17,9003) XY,' MIN 0.0, MAX 100.0.'
     ELSEIF ((VALUE .EQ. 'G' ) .OR. ( VALUE .EQ. 'g')) THEN
      LABEL = 'd'
      RANGE = 1000.0
      MINVAL = 1000.0
      WRITE(17,9003) XY,' LABEL "Depth ((m))".'
      WRITE(17,9003) XY,' MIN 1000.0, MAX 2000.0.'
      UA = .TRUE.
70    WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2  'LAYER (XXXX.):'
      READ(4,9021) ULAYER
      WRITE(6,9003) 'ENTER THE PRESSURE OF THE LOWER ',
     2  'LAYER (XXXX.)'
      WRITE(6,9003) '     A NEGATIVE ENTRY YEILDS THE ',
     2  'SURFACE VALUE:'
      READ(4,9021) LLAYER
      IF ( ULAYER .LT. ULAYER ) GOTO 70
     ELSEIF ((VALUE .EQ. 'H' ) .OR. ( VALUE .EQ. 'h')) THEN
      LABEL = 'SFC Pressure'
      RANGE = 15.0
      MINVAL = 85.0
      WRITE(17,9003) XY,' LABEL "SFC Pressure ((kPa))".'
      WRITE(17,9003) XY,' MIN 85.0, MAX 100.0.'

     ELSEIF ((VALUE .EQ. 'I' ) .OR. ( VALUE .EQ. 'i')) THEN
      LABEL = 'RH'
      MINVAL = 60.0
      RANGE= 40.0
      WRITE(17,9003) XY,' LABEL "RH".'
      WRITE(17,9003) XY,' MIN 40.0, MAX 100.0.'
     ELSEIF ((VALUE .EQ. 'J' ) .OR. ( VALUE .EQ. 'j')) THEN
      LABEL = 'Tw'
      MINVAL = -6.0
      RANGE= 12.0
      WRITE(17,9003) XY,' LABEL "Tw".'
      WRITE(17,9003) XY,' MIN -6.0, MAX 6.0.'
     ELSEIF ((VALUE .EQ. 'K' ) .OR. ( VALUE .EQ. 'k')) THEN
      LABEL = 'Lapse Rate'
      MINVAL = -15.0
```

```
      RANGE = 20.0
      WRITE(17,9003) XY,' LABEL "Lapse Rate ((C/km))".'
      WRITE(17,9003) XY,' MIN -15.0, MAX 5.0.'
      UA = .TRUE.
120   WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2 'LAYER (XXXX.):'
      READ(4,9021) ULAYER
      WRITE(6,9003) 'ENTER THE PRESSURE OF THE LOWER ',
     2 'LAYER (XXXX.)'
      WRITE(6,9003) '     A NEGATIVE ENTRY YEILDS THE ',
     2 'SURFACE VALUE:'
      READ(4,9021) LLAYER
      IF ( ULAYER .LT. ULAYER ) GOTO 120
     ELSEIF ((VALUE .EQ. 'L' ) .OR. ( VALUE .EQ. 'l')) THEN
      LABEL = 'Wind Shear'
      MINVAL = -15.0
      RANGE = 20.0
      WRITE(17,9003) XY,' LABEL "Wind Shear ((1/C))".'
      WRITE(17,9003) XY,' MIN -15.0, MAX 5.0.'
      UA = .TRUE.
130   WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2 'LAYER (XXXX.):'
      READ(4,9021) ULAYER
      WRITE(6,9003) 'ENTER THE PRESSURE OF THE LOWER ',
     2 'LAYER (XXXX.)'
      WRITE(6,9003) '     A NEGATIVE ENTRY YEILDS THE ',
     2 'SURFACE VALUE:'
      READ(4,9021) LLAYER
      IF ( ULAYER .LT. ULAYER ) GOTO 130
     ELSEIF ((VALUE .EQ. 'M' ) .OR. ( VALUE .EQ. 'm')) THEN
      LABEL = 'Change in Ta'
      MINVAL = -20.0
      RANGE = 25.0
      WRITE(17,9003) XY,' LABEL "Change in Ta ((C))".'
      WRITE(17,9003) XY,' MIN -20.0, MAX 5.0.'
      UA = .TRUE.
140   WRITE(6,9003) 'ENTER THE PRESSURE OF THE UPPER ',
     2 'LAYER (XXXX.):'
      READ(4,9021) ULAYER
      WRITE(6,9003) 'ENTER THE PRESSURE OF THE LOWER ',
     2 'LAYER (XXXX.)'
      WRITE(6,9003) '     A NEGATIVE ENTRY YEILDS THE ',
     2 'SURFACE VALUE:'
      READ(4,9021) LLAYER
      IF ( ULAYER .LT. ULAYER ) GOTO 140
     ELSEIF ((VALUE .EQ. 'N' ) .OR. ( VALUE .EQ. 'n')) THEN
      UA = .TRUE.
      LABEL = 'UA P((SFC))'
      MINVAL = 90.0
      RANGE = 10.0
      ULAYER = -1
      WRITE(17,9003) XY,' LABEL "Surface Pressure ',
     2 '((kPa))".'
      WRITE(17,9003) XY,' MIN 90.0, MAX 100.0.'
```

```
      ELSEIF ((VALUE .EQ. 'O' ) .OR. ( VALUE .EQ. 'o')) THEN
         LABEL = 'Wind Direction'
         MINVAL = 0.0
         RANGE = 360.0
         WRITE(17,9003) XY,' LABEL "Sfc Wind Direction ',
     2 '((m/s))".'
         WRITE(17,9003) XY,' MIN 00.0, MAX 360.0.'
      ELSEIF ((VALUE .EQ. 'P' ) .OR. ( VALUE .EQ. 'p')) THEN
         LABEL = 'Sfc Wind Speed'
         MINVAL = 0.0
         RANGE = 40.0
         WRITE(17,9003) XY,' LABEL "Sfc Wind Speed ',
     2 '((kPa))".'
         WRITE(17,9003) XY,' MIN 0.0, MAX 40.0.'
      ELSEIF ((VALUE .EQ. 'Q' ) .OR. ( VALUE .EQ. 'q')) THEN
         LABEL = 'Visibility'
         MINVAL = 0.0
         RANGE = 20.0
         WRITE(17,9003) XY,' LABEL "Visibility ((km))".'
         WRITE(17,9003) XY,' MIN 0.0, MAX 20.0.'
      ELSEIF ((VALUE .EQ. 'R' ) .OR. ( VALUE .EQ. 'r')) THEN
         LABEL = 'Hourly Precipitation'
         MINVAL = 0.0
         RANGE = 20.0
         WRITE(17,9003) XY,' LABEL "Hourly Precipitation ',
     2   '((mm Water))".'
         WRITE(17,9003) XY,' MIN 0.0, MAX 20.0.'
      ELSEIF ( ( VALUE .EQ. 'Z' ) .OR.
     2 ( VALUE .EQ. 'z' ) ) THEN
         LABEL = 'Unity'
         WRITE(17,9003) ' '
         WRITE(17,9003) ' '
      ELSE
         GOTO 55
      ENDIF

 9001 FORMAT(A)
 9002 FORMAT(2X,3(I2,3X),1X,I2,2X,F8.1,10X,3(2X,F8.1),12X,
     2 F8.1)
 9003 FORMAT(1X,A,A,A,A)
 9004 FORMAT(I2)
 9005 FORMAT(1X,F8.2,A,F8.2)
 9006 FORMAT(1X,A,I2,A)
 9007 FORMAT(1X,A,I2,A,I2,1X,A)
 9009 FORMAT(F4.2)
 9018 FORMAT(1X,A,A,A,A,A,I2,A,I2,A)
 9021 FORMAT(F5.0)
 9022 FORMAT(I1)
 9999 RETURN
C     end of subroutine VARABS
      END
```

7.5.1.8 Subroutine PREAVE

```
C  SUBROUTINE  ************************************************
C                                                              *
C    Purpose: to store the all the variables in the matrix     *
C         PTS, and to count the number of data (x,y)           *
C         points. Indices are:                                 *
C              1) the x-variable,                              *
C              2) the y-variable,                              *
C              3) the duration of the precipitation event      *
C              associated with the x and y variables, or       *
C              the time from the onset of the                  *
C              precipitation event.                            *
C                                                              *
C    Definitions:                                              *
C         ALLPTS - see METSTAT                                 *
C         K      - the duration of the precipitation event     *
C                  associated with the x and y variables, or   *
C                  the time (of the x and y-variables) from    *
C                  the onset of the precipitation event.       *
C         PTS    - see METSTAT                                 *
C         XVAL   - see METSTAT                                 *
C         YVAL   - see METSTAT                                 *
C                                                              *
C    Programmed by Mark Bourassa  1988                         *
C                                                              *
C**************************************************************

      SUBROUTINE PREAVE( ALLPTS, PTS, XVAL, YVAL, K )

      INTEGER ALLPTS(2), K
      REAL PTS(1600,3), XVAL, YVAL

      ALLPTS(1) = ALLPTS(1) + 1
      PTS(ALLPTS(1),1) = XVAL
      PTS(ALLPTS(1),2) = YVAL
      PTS(ALLPTS(1),3) = REAL(K)

      RETURN
      END
```

7.5.1.9 Subroutine AVERAG

```
C  SUBROUTINE AVERAG  ******************************************
C                                                              *
C    Purpose: if there is only one variable, to find its       *
C         mean and standard deviation for each hour of         *
C         interest and its mean and standard deviation for     *
C         whole data set. If there are two variables the       *
C         purpose is to determine the best fit line for        *
C         a linear relationship between the two variables.     *
C                                                              *
```

```
C     Definitions:                                                    *
C          ALLPTS - see METSTAT                                       *
C          AVE    - average                                           *
C          DVDND  - the dividend in the equations for a               *
C                   least squares linear fit                          *
C          JMAX   - see METSTAT                                       *
C          JMIN   - see METSTAT                                       *
C          LGCHIS - see METSTAT                                       *
C          LGCNRM - see METSTAT                                       *
C          LGCZRO - see METSTAT                                       *
C          PCENT  - the percentage of values that are equal           *
C                   to zero for a variable                            *
C          PTS    - see METSTAT                                       *
C          R      - correlation coefficient                           *
C          SIGMAB - standard deviation in the y-intercept             *
C          SIGMAM - standard deviation in the x-intercept             *
C          SIGMAY - standard deviation in y of the best fit           *
C                   line                                              *
C          SLOPE  - slope of the best fit line                        *
C          SUMX   - sum of the x-values                               *
C          SUMX2  - sum of the squares of the x-values                *
C          SUMXY  - sum of the product of the x and                   *
C                   y-values                                          *
C          SUMY   - sum of the y-values                               *
C          SUMY2  - sum of the squares of the y-values                *
C          TRUPTS - array of the number of data points for            *
C                   each hour from the onset of the event             *
C          VARXY  - covariance                                        *
C          XMEAN  - mean x-values                                     *
C          XVALUE - see METSTAT                                       *
C          XVAR   - variance in x                                     *
C          YINT   - y-intercept of the best fit line for              *
C                   the linear relationship between x and y           *
C          YMEAN  - mean y-values                                     *
C          YVALUE - see METSTAT                                       *
C          YVAR   - variance in y                                     *
C          ZERO   - the number of values that are equal to            *
C                   zero. If there are two variables then             *
C                   ZERO is the number of times both are              *
C                   equal to zero                                     *
C                                                                     *
C     Programmed by Mark Bourassa   1988                              *
C                                                                     *
C*********************************************************************

      SUBROUTINE AVERAG( JMIN, JMAX, ALLPTS, PTS, XVALUE,
     2 YVALUE, LGCZRO, LGCHIS, LGCNRM )

      LOGICAL LGCZRO, LGCHIS, LGCNRM
      INTEGER JMIN, JMAX, ALLPTS, PCENT(61)
      REAL SUMX(61), SUMX2(61), SUMY(61), SUMXY(61),
     + TRUPTS(61), DVDND(61), SLOPE(61), SIGMAM(61),
     + YINT(61), SIGMAB(61), PTS(1600,3), AVE(61),
     + SUMY2(61), SIGMAY(61), ZERO(61), XVAR(61), YVAR(61),
```

```
     + VARXY(61), XMEAN(61), YMEAN(61), R(61)
       CHARACTER*1 XVALUE, YVALUE

       DATA SUMX/61*0.0/
       DATA SUMY/61*0.0/
       DATA SUMX2/61*0.0/
       DATA SUMXY/61*0.0/
       DATA SUMY2/61*0.0/
       DATA TRUPTS/61*0.0/
       DATA DVDND/61*0.0/
       DATA SLOPE/61*0.0/
       DATA SIGMAM/61*0.0/
       DATA YINT/61*0.0/
       DATA SIGMAB/61*0.0/
       DATA AVE/61*0.0/
       DATA SIGMAY/61*0.0/
       DATA ZERO/61*0.0/
       DATA XMEAN/61*0.0/
       DATA YMEAN/61*0.0/
       DATA VARXY/61*0.0/
       DATA XVAR/61*0.0/
       DATA YVAR/61*0.0/

       DO 100 K=1,ALLPTS
       J = INT( PTS(K,3) )
          IF ( ( PTS(K,1) .EQ. 0.0 ) .AND.
     2    ( PTS(K,2) .EQ. 0.0 ) .AND. LGCZRO ) THEN
             ZERO( J ) = ZERO( J ) + 1.0
             ZERO(61) = ZERO(61) + 1.0
          ELSE
             SUMX( J ) = SUMX( J ) + PTS(K,1)
             SUMY( J ) = SUMY( J ) + PTS(K,2)
             SUMX2( J ) = SUMX2( J ) + PTS(K,1) * PTS(K,1)
             SUMXY( J ) = SUMXY( J ) + PTS(K,1) * PTS(K,2)
             SUMY2( J ) = SUMY2( J ) + PTS(K,2) * PTS(K,2)
             TRUPTS( J ) = TRUPTS( J ) + 1.0
             SUMX( 61 ) = SUMX( 61 ) + PTS(K,1)
             SUMY( 61 ) = SUMY( 61 ) + PTS(K,2)
             SUMX2( 61 ) = SUMX2( 61 ) + PTS(K,1) * PTS(K,1)
             SUMXY( 61 ) = SUMXY( 61 ) + PTS(K,1) * PTS(K,2)
             SUMY2( 61 ) = SUMY2( 61 ) + PTS(K,2) * PTS(K,2)
             TRUPTS( 61 ) = TRUPTS( 61 ) + 1.0
          ENDIF
   100 CONTINUE

       IF ( LGCZRO ) THEN
          DO 125 I=1,60
             IF ( ( TRUPTS(I) + ZERO(I) ) .NE. 0.0 ) THEN
                PCENT(I) = INT( 100.0 * ZERO(I) /
     2          ( TRUPTS(I) + ZERO(I) ) )
             ELSE
                PCENT(I) = -1.0
             ENDIF
   125    CONTINUE
```

```
      PCENT(61) = INT( 100.0 * ZERO(61) / ALLPTS )
      DO 135 I=0,5
         WRITE(23,9002) 'Hour', (J,J=10*I+1,10*(I+1))
         WRITE(23,9002) 'Null change (%)', (PCENT(J),
     2   J=10*I+1,10*(I+1))
         WRITE(23,9001) '  '
  135 CONTINUE
      WRITE(23,9023) PCENT(61), 'percent of all changes',
     2 ' were null changes.'
      ENDIF

      IF (((XVALUE .NE. 'Z') .AND. (XVALUE .NE. 'z')) .AND.
     2 ( (YVALUE .NE. 'Z') .AND. (YVALUE .NE. 'z' ) ) ) THEN
      DO 1000 K = JMIN, JMAX
         DVDND(K) = TRUPTS(K) * SUMX2(K) - SUMX(K) *
     2   SUMX(K)
         IF ( ( DVDND(K) .NE. 0.0 ) .AND.
     2   ( TRUPTS(K) .NE. 1.0 ) ) THEN
            SLOPE(K) = (TRUPTS(K) * SUMXY(K) - SUMX(K) *
     2      SUMY(K)) / DVDND(K)
            YINT(K) = ( SUMX2(K) * SUMY(K) - SUMX(K) *
     2      SUMXY(K) ) / DVDND(K)
         ENDIF
 1000 CONTINUE
      DVDND(61) = TRUPTS(61) * SUMX2(61) - SUMX(61) *
     2 SUMX(61)
      IF ( ( DVDND(61) .NE. 0.0 ) .AND.
     2 ( TRUPTS(61) .NE. 1.0 ) ) THEN
         SLOPE(61) = (TRUPTS(61) * SUMXY(61) - SUMX(61) *
     2   SUMY(61)) / DVDND(61)
         YINT(61) = ( SUMX2(61) * SUMY(61) - SUMX(61) *
     2   SUMXY(61) ) / DVDND(61)
      ENDIF
      DO 1500 I=1, ALLPTS
         K = INT( PTS(I,3) )
         SIGMAY(K) = SIGMAY(K) + ( PTS(I,2) - YINT(K) -
     2   SLOPE(K) * PTS(I,1) )**2
         SIGMAY(61) = SIGMAY(61) + ( PTS(I,2) - YINT(61)
     2   - SLOPE(61) * PTS(I,1) )**2
 1500 CONTINUE
      DO 2000 K=JMIN, JMAX
         IF ( ( INT( TRUPTS(K) ) .GT. 2 ) .AND.
     2   ( DVDND(K) .NE. 0.0 ) ) THEN
            SIGMAY(K) = SQRT( SIGMAY(K) /
     2      ( REAL( TRUPTS(K) ) - 2.0 ) )
            SIGMAB(K) = SQRT( SIGMAY(K) *
     2      SIGMAY(K) * SUMX2(K) / DVDND(K) )
            SIGMAM(K) = SQRT( REAL( TRUPTS(K) ) *
     2      SIGMAY(K) * SIGMAY(K) / DVDND(K) )
         ENDIF
 2000 CONTINUE
      IF ( ( INT( TRUPTS(61) ) .GT. 2 ) .AND.
     2 ( DVDND(61) .NE. 0.0 ) ) THEN
         SIGMAY(61) = SQRT( SIGMAY(61) /
```

```
     2       ( REAL( TRUPTS(61) ) - 2.0 ) )
             SIGMAB(61) = SQRT( SIGMAY(61) *
     2       SIGMAY(61) * SUMX2(61) / DVDND(61) )
             SIGMAM(61) = SQRT( REAL( TRUPTS(61) ) *
     2       SIGMAY(61) * SIGMAY(61) / DVDND(61) )
          ENDIF

          DO 2100 I = JMIN, JMAX
             XMEAN(I) = SUMX(I) / TRUPTS(I)
             YMEAN(I) = SUMY(I) / TRUPTS(I)
2100      CONTINUE
          XMEAN(61) = SUMX(61) / TRUPTS(61)
          YMEAN(61) = SUMY(61) / TRUPTS(61)
          DO 2200 I = 1, ALLPTS
             VARXY( PTS(I,3) ) = VARXY( PTS(I,3) ) +
     2       ( PTS(I,1) - XMEAN( PTS(I,3) ) ) *
     3       ( PTS(I,2) - YMEAN( PTS(I,3) ) )
             XVAR( PTS(I,3) ) = XVAR( PTS(I,3) ) +
     2       ( PTS(I,1) - XMEAN( PTS(I,3) ) ) ** 2
             YVAR( PTS(I,3) ) = YVAR( PTS(I,3) ) +
     2       ( PTS(I,2) - YMEAN( PTS(I,3) ) ) ** 2
             VARXY( 61 ) = VARXY(61) +
     2       (PTS(I,1) - XMEAN(61)) * (PTS(I,2) - YMEAN(61))
             XVAR(61) = XVAR(61) + (PTS(I,1) - XMEAN(61)) **2
             YVAR(61) = YVAR(61) + (PTS(I,2) - YMEAN(61)) **2
2200      CONTINUE
          DO 2300 I=JMIN, JMAX
             IF ( TRUPTS(I) .GT. 2 ) THEN
                R(I) = VARXY(I) / SQRT( XVAR(I) * YVAR(I) )
             ELSE
                R(I) = -9.99
             ENDIF
2300      CONTINUE
          IF ( TRUPTS(61) .GT. 2 ) THEN
             R(61) = VARXY(61) / SQRT( XVAR(61) * YVAR(61) )
          ELSE
             R(61) = -9.99
          ENDIF

          WRITE(22,9021) 'Time', (K,K=JMIN,JMAX)
          WRITE(22,9022) 'Slope      ',
     2    (SLOPE(K),K=JMIN,JMAX), SLOPE(61)
          WRITE(22,9022) 'Stand. Dev.',
     2    (SIGMAM(K),K=JMIN,JMAX), SIGMAM(61)
          WRITE(22,9022) 'Y-Intercept',
     2    (YINT(K),K=JMIN,JMAX), YINT(61)
          WRITE(22,9022) 'Stand. Dev.',
     2    (SIGMAB(K),K=JMIN,JMAX), SIGMAB(61)
          DO 2400 I=1,61
             IF ( ( TRUPTS(I) + ZERO(I) ) .NE. 0.0 ) THEN
                ZERO(I) = INT( 100.0 * ZERO(I) /
     2          ( TRUPTS(I) + ZERO(I) ) )
             ELSE
                ZERO(I) = -1.0
```

```
            ENDIF
2400      CONTINUE
          PCENT(61) = INT( 100.0 * ZERO(61) / ALLPTS )
       WRITE(22,9022) '% Null Diff',
     2    (ZERO(K),K=JMIN,JMAX), ZERO(61)
          WRITE(22,9022) 'Corr. Coef.', (R(K),K=JMIN,JMAX),
     2    R(61)
          WRITE(22,9021) 'Data Points',
     2    (INT(TRUPTS(K)),K=JMIN,JMAX),INT( TRUPTS(61) )
        ELSEIF ( ( XVALUE .NE. 'Z' ) .AND.
     2  ( XVALUE .NE. 'z' ) ) THEN
          DO 2500 K=JMIN, JMAX
             IF ( TRUPTS(K) .GT. 1.0 ) THEN
                AVE(K) = SUMX(K) / TRUPTS(K)
                SIGMAY(K) = ( SUMX2(K) - TRUPTS(K) * AVE(K) *
     2          AVE(K) ) / ( TRUPTS(K) - 1.0 )
                IF ( SIGMAY(K) .NE. 0.0 ) SIGMAY(K) =
     2          SQRT( SIGMAY(K) )
             ENDIF
2500      CONTINUE
          IF ( TRUPTS(61) .GT. 0.0 ) THEN
             AVE(61) = SUMX(61)/TRUPTS(61)
             SIGMAY(61) = ( SUMX2(61) - TRUPTS(61) * AVE(61)
     2       * AVE(61) ) / ( TRUPTS(61) - 1.0 )
             SIGMAY(61) = SQRT( SIGMAY(61) )
          ENDIF
          WRITE(22,9021) 'Time', (K,K=JMIN,JMAX)
          WRITE(22,9022) 'Average ', (AVE(K),K=JMIN,JMAX),
     2    AVE(61)
          WRITE(22,9022) 'Stand. Dev.',
     2    (SIGMAY(K),K=JMIN,JMAX), SIGMAY(61)
          DO 2600 I=1,61
             IF ( ( TRUPTS(I) + ZERO(I) ) .NE. 0.0 ) THEN
                ZERO(I) = INT( 100.0 * ZERO(I) /
     2          ( TRUPTS(I) + ZERO(I) ) )
             ELSE
                ZERO(I) = -1.0
             ENDIF
2600      CONTINUE
          PCENT(61) = INT( 100.0 * ZERO(61) / ALLPTS )
       WRITE(22,9022) '% Null Diff',
     2    (ZERO(K),K=JMIN,JMAX),ZERO(61)
          WRITE(22,9021) 'Points', (INT( TRUPTS(K) ),
     2    K=JMIN,JMAX),INT( TRUPTS(61) )

          IF ( LGCHIS) CALL HISTO( LGCZRO, LGCNRM, ALLPTS, 1,
     2    JMIN, JMAX, PTS, AVE, SIGMAY )

       ELSE
          DO 3000 K=JMIN, JMAX
             IF ( TRUPTS(K) .GT. 1.0 ) THEN
                AVE(K) = SUMY(K) / TRUPTS(K)
                SIGMAY(K) = ( SUMY2(K) - TRUPTS(K) * AVE(K) *
     2          AVE(K) ) / ( TRUPTS(K) - 1.0 )
```

```
                IF ( SIGMAY(K) .NE. 0.0 ) SIGMAY(K) =
     2          SQRT( SIGMAY(K) )
             ENDIF
3000      CONTINUE
          IF ( TRUPTS(61) .GT. 1.0 ) THEN
             AVE(61) = SUMY(61) / TRUPTS(61)
             SIGMAY(61) = ( SUMY2(61) - TRUPTS(61) * AVE(61)
     2       * AVE(61) ) / ( TRUPTS(61) - 1.0 )
             SIGMAY(61) = SQRT( SIGMAY(61) )
          ENDIF
          WRITE(22,9021) 'Time', (K,K=JMIN,JMAX)
          WRITE(22,9022) 'Average    ', (AVE(K),K=JMIN,JMAX),
     2    AVE(61)
          WRITE(22,9022) 'Stand. Dev.',
     2    (SIGMAY(K),K=JMIN,JMAX), SIGMAY(61)
          DO 3200 I=1,61
             IF ( ( TRUPTS(I) + ZERO(I) ) .NE. 0.0 ) THEN
                ZERO(I) = INT( 100.0 * ZERO(I) /
     2          ( TRUPTS(I) + ZERO(I) ) )
             ELSE
                ZERO(I) = -1.0
             ENDIF
3200      CONTINUE
          PCENT(61) = INT( 100.0 * ZERO(61) / ALLPTS )
         WRITE(22,9022) '% Null Diff',
     2    (ZERO(K),K=JMIN,JMAX),ZERO(61)
          WRITE(22,9021) 'Points', (INT( TRUPTS(K) ),
     2    K=JMIN,JMAX), INT( TRUPTS(61) )

          IF ( LGCHIS) CALL HISTO( LGCZRO, LGCNRM, ALLPTS, 2,
     2    JMIN, JMAX, PTS, AVE, SIGMAY )

         ENDIF

 9001 FORMAT(1X,A,A,A)
 9002 FORMAT(1X,A,T25,10I3)
 9021 FORMAT(1X,A,7X,61(3X,I3,2X))
 9022 FORMAT(1X,A,61(1X,F7.3))
 9023 FORMAT(1X,I3,1X,A,A,A)

         RETURN
C        end of subroutine AVERAG
         END
```

7.5.1.10 Subroutine HISTO

```
C  SUBROUTINE HISTO  ****************************************
C                                                           *
C    Purpose: to make a histogram of one of the frequency   *
C         distribution of one varaible (the other variable  *
C         must be unity). The distribution is in z-units    *
C         (standard deviaitons). The histogram does not     *
C         have to be normalized (LGCNRM). A chi squared     *
```

```
C         test is performed on the distribution, to          *
C         determine how will it fits a Gaussian              *
C         distribution.                                      *
C                                                            *
C     Definitions:                                           *
C          ALLPTS - see METSTAT                              *
C          CHI2   - reduced chi squared value for the        *
C                   distributions goodness of fit to a       *
C                   Gaussian distribution                    *
C          HISTOG - counters for each pool in histogram.     *
C                   Each pool is a qwuater of a standard     *
C                   deviaition wide                          *
C          I      - counter                                  *
C          J      - counter                                  *
C          K      - counter                                  *
C          LGCNRM - see METSTAT                              *
C          LGCZRO - see METSTAT                              *
C          MAX    - the greatest duration or time from       *
C                   onset of interest                        *
C          MAXSDV - the upper range of the histogram,        *
C                   measured in quarter standard deviations  *
C          MEAN   - the mean of the variable                 *
C          MIN    - the least duration or time from onset    *
C                   of interest                              *
C          MINSDV - the lower range of he histogram,         *
C                   measured in quarter standard deviations  *
C          OBSR   - a set of twelve data pools, bounded in   *
C                   terms of standard deviations from the    *
C                   mean. If the distribution is Gaussian    *
C                   then the size of the pools should be     *
C                   equal                                    *
C          PTS    - see METSTAT                              *
C          SIGMA  - the standard deviation of the variable   *
C          XY     - see AVERAG                               *
C          ZDEV   - number of standard deviations            *
C                                                            *
C     Programmed by Mark Bourassa   1988                     *
C                                                            *
C************************************************************

      SUBROUTINE HISTO( LGCZRO, LGCNRM, ALLPTS, XY, MIN,
     2 MAX, PTS, MEAN, SIGMA )

      LOGICAL LGCZRO, LGCNRM
      INTEGER ALLPTS, XY, MIN, MAX, I, J, MINSDV, MAXSDV, K
      REAL PTS(1600,3), MEAN(61), SIGMA(61), HISTOG(61,25),
     2 ZDEV, OBSR(13,61), CHI2(61)

      DATA HISTOG/1525*0.0/
      DATA OBSR/793*0.0/
      DATA CHI2/61*0.0/
      MINSDV = -8
      MAXSDV = 8
```

```
      DO 100 I=1, ALLPTS
      K = INT( PTS(I,3) )
         IF ( SIGMA( K ) .GT. 0.0 ) THEN
            IF ( .NOT. ( ( PTS(I,XY) .EQ. 0.0 ) .AND. LGCZRO
     2      ) ) THEN
C     determine the number of standard deviations from the
C     mean, and increment the total and the appropriate pool
               ZDEV = ( PTS(I,XY) - MEAN(K) ) / SIGMA( K )
               IF ( ZDEV .LE. -1.3844) THEN
                  OBSR(1,K) = OBSR(1,K) + 1.0
                  OBSR(1,61) = OBSR(1,61) + 1.0
               ELSEIF ( ( ZDEV .GT. -1.3844 ) .AND.
     2         ( ZDEV .LE. -0.9773 ) ) THEN
                  OBSR(2,K) = OBSR(2,K) + 1.0
                  OBSR(2,61) = OBSR(2,61) + 1.0
               ELSEIF ( ( ZDEV .GT. -0.9773 ) .AND.
     2         ( ZDEV .LE. -0.6745 ) ) THEN
                  OBSR(3,K) = OBSR(3,K) + 1.0
                  OBSR(3,61) = OBSR(3,61) + 1.0
               ELSEIF ( ( ZDEV .GT. -0.6745 ) .AND.
     2         ( ZDEV .LE. -0.4316 ) ) THEN
                  OBSR(4,K) = OBSR(4,K) + 1.0
                  OBSR(4,61) = OBSR(4,61) + 1.0
               ELSEIF ( ( ZDEV .GT. -0.4316 ) .AND.
     2         ( ZDEV .LE. -0.2152 ) ) THEN
                  OBSR(5,K) = OBSR(5,K) + 1.0
                  OBSR(5,61) = OBSR(5,61) + 1.0
               ELSEIF ((ZDEV .GT. -0.2152) .AND.
     2         (ZDEV .LE. 0.00)) THEN
                  OBSR(6,K) = OBSR(6,K) + 1.0
                  OBSR(6,61) = OBSR(6,61) + 1.0
               ELSEIF (( ZDEV .GT. 0.00) .AND.
     2         (ZDEV .LE. 0.2152)) THEN
                  OBSR(7,K) = OBSR(7,K) + 1.0
                  OBSR(7,61) = OBSR(7,61) + 1.0
               ELSEIF ((ZDEV .GT. 0.2152) .AND.
     2         (ZDEV .LE. 0.4316)) THEN
                  OBSR(8,K) = OBSR(8,K) + 1.0
                  OBSR(8,61) = OBSR(8,61) + 1.0
               ELSEIF ((ZDEV .GT. 0.4316) .AND.
     2         (ZDEV .LT. 0.6745)) THEN
                  OBSR(9,K) = OBSR(9,K) + 1.0
                  OBSR(9,61) = OBSR(9,61) + 1.0
               ELSEIF ((ZDEV .GT. 0.6745) .AND.
     2         (ZDEV .LT. 0.9773)) THEN
                  OBSR(10,K) = OBSR(10,K) + 1.0
                  OBSR(10,61) = OBSR(10,61) + 1.0
               ELSEIF ((ZDEV .GT. 0.9773) .AND.
     2         (ZDEV .LT. 1.3844)) THEN
                  OBSR(11,K) = OBSR(11,K) + 1.0
                  OBSR(11,61) = OBSR(11,61) + 1.0
               ELSE
                  OBSR(12,K) = OBSR(12,K) + 1.0
                  OBSR(12,61) = OBSR(12,61) + 1.0
```

```
                ENDIF
                OBSR(13,K) = OBSR(13,K) + 1.0
                POSN = INT( 4.0 * ZDEV ) + 13
                IF ( ( POSN .GE. 1 ) .AND. ( POSN .LE. 24 ) )
     2          HISTOG( K, POSN ) = HISTOG( K, POSN ) +1.0
                HISTOG( K, 25 ) = HISTOG( K, 25 ) + 1.0
             ENDIF
             IF ( POSN .LT. 5 ) MINSDV = -12
             IF ( POSN .GT. 21 ) MAXSDV = 12
          ENDIF
  100 CONTINUE

C     determine the reduced chi squared values
      DO 110 I=1,12
         DO 105 J=MIN,MAX
            IF ( OBSR(13,J) .GT. 0.0 )  CHI2(J) = CHI2(J) +
     2      1.0909 * ( ( OBSR(I,J) - OBSR(13,J) / 12.0 )**2)
     3      / OBSR(13,J)
            OBSR(13,61) = OBSR(13,61) + OBSR(I,J)
  105    CONTINUE
         CHI2(61) = CHI2(61) + 1.0909 *
     2   ( ( OBSR(I,61) - OBSR(13,61) / 12.0 ) **.2) /
     3   OBSR(13,61)
  110 CONTINUE

      DO 150 J=1,25
         DO 125 I=MIN,MAX
            HISTOG(61,J) = HISTOG(61,J) + HISTOG(I,J)
  125    CONTINUE
  150 CONTINUE

C     if the histogram isto be normalized then do so
      IF ( LGCNRM ) THEN
         DO 300 I=1,61
            DO 200 J=1,25
               IF ( HISTOG(I,25) .GT. 0.0 ) THEN
                  HISTOG(I,J) = ( HISTOG(I,J) + 0.0005 )
     2            / HISTOG(I,25)
               ELSE
                  HISTOG(I,J) = 0.0
               ENDIF
  200       CONTINUE
  300    CONTINUE
      ENDIF

      WRITE(22,9022) 'Red. Chi**2', (CHI2(K),K=MIN,MAX),
     2 CHI2(61)

     DO 102 J=MIN, MAX
          WRITE(22,9981) (OBSR(I,J),I=1,13)
 9981     FORMAT( 2X, 13(F5.1,1X) )
  102     CONTINUE
     WRITE(22,9981) (OBSR(I,61),I=1,13)
```

```
      WRITE(24,9001) 'Distance from the mean (in standard ',
     2 'deviations)'
      WRITE(24,9002) ' From ',
     2 (REAL(J)/4.0,J=MINSDV,MAXSDV-1)
      WRITE(24,9002) '  To  ',
     2 (REAL(J)/4.0,J=MINSDV+1,MAXSDV)
           600 I=MIN, MAX
           WRITE(24,9002) 'Points', (HISTOG(I,J),
     2     J=MINSDV + 13, MAXSDV + 13)
   600     CONTINUE
           WRITE(24,9002) 'Points', (HISTOG(61,J),
     2     J=MINSDV + 13,MAXSDV + 13)
           WRITE(24,9002) ' From ', (REAL(J)/4.0,J=MINSDV,
     2     MAXSDV-1)
           WRITE(24,9002) '  To  ', (REAL(J)/4.0,J=MINSDV+1,
     2     MAXSDV)

 9001 FORMAT( 1X,A,A )
 9002 FORMAT( 1X,A,1X,25(F6.3,1X))
 9003 FORMAT(A)
 9022 FORMAT(1X,A,61(2X,F7.3))

      RETURN
C     end of subroutine HISTO
      END
```

7.5.1.11 Subroutine MINIMA

```
C  SUBROUTINE MINIMA  ****************************************
C                                                             *
C    Purpose: to find the minimum (or maximum) y-value in     *
C         each of the x-value data pools. These maxima        *
C         will be used in subrotuine STATS to find the        *
C         best fit line for a linear relationship between     *
C         the minimum (or maximum) y-values as a function     *
C         of the x-variable.                                  *
C                                                             *
C    Definitions:                                             *
C         ALLPTS - see METSTAT                                *
C         DPTS   - see METSTAT                                *
C         K      - either the duration of the                 *
C                  precipitation event, or the number of      *
C                  hours since the onset of the event         *
C         MINPTS - see METSTAT                                *
C         MINVAL - see METSTAT                                *
C         POOL   - the index of MINVAL that represent the     *
C                  postion of the data pool                   *
C         RANGE  - see METSTAT                                *
C         S      - see METSTAT                                *
C         XVAL   - see METSTAT                                *
C         YVAL   - see METSTAT                                *
C                                                             *
C    Programmed by Mark Bourassa   1988                       *
```

```
C                                                                      *
C**********************************************************************

      SUBROUTINE MINIMA( MINPTS, XVAL, YVAL, DPTS, K, S,
     2 RANGE, MINVAL )

      INTEGER K, DPTS, POOL, ALLPTS
      REAL MINPTS(100,2,60), XVAL, YVAL, S, RANGE, MINVAL

      POOL = INT( REAL( DPTS ) * ( XVAL - MINVAL ) / RANGE )
      IF ( S * YVAL .LT. MINPTS( POOL,2,K )  ) THEN
         MINPTS(POOL,1,K) = XVAL
         MINPTS(POOL,2,K) = YVAL
      ENDIF

      RETURN
C     end of subroutine MINIMA
      END
```

7.5.1.11 Subroutine STATS

```
C  SUBROUTINE STATS   ************************************************
C                                                                      *
C    Purpose: to find the best fit line for a linear                   *
C         relationship between the minimum (or maximum)                *
C         y-values as a function of the x-variable.                    *
C                                                                      *
C    Definitions:                                                      *
C         DPTS    - see METSTAT                                        *
C         DVDND   - see AVERAG                                         *
C         I       - counter                                            *
C         JMAX    - see METSTAT                                        *
C         JMIN    - see METSTAT                                        *
C         K       - counter                                            *
C         L       - counter                                            *
C         MAX     - counter for the maximum hour of                    *
C                   interest                                           *
C         MIN     - counter for the minimum hour of                    *
C                   interest                                           *
C         MINPTS  - see METSTAT                                        *
C         PTS     - see METSTAT                                        *
C         R       - see AVERAG                                         *
C         S       - see METSTAT                                        *
C         SIGMAB  - see AVERAG                                         *
C         SIGMAM  - see AVERAG                                         *
C         SIGMAY  - see AVERAG                                         *
C         SLOPE   - see AVERAG                                         *
C         SUMX    - see AVERAG                                         *
C         SUMX2   - see AVERAG                                         *
C         SUMXY   - see AVERAG                                         *
C         SUMY    - see AVERAG                                         *
C         TRUPTS  - see AVERAG                                         *
C         VARXY   - see AVERAG                                         *
```

```
C         XMEAN   - see AVERAG                                          *
C         XVAR    - see AVERAG                                          *
C         YINT    - see AVERAG                                          *
C         YMEAN   - see AVERAG                                          *
C         YVAR    - see AVERAG                                          *
C                                                                       *
C     Programmed by Mark Bourassa  1988                                 *
C                                                                       *
C***********************************************************************

      SUBROUTINE STATS( MINPTS,DPTS,S,JMIN,JMAX )

      INTEGER DPTS, TRUPTS, MIN, MAX, I , JMIN, JMAX, K, L
      REAL MINPTS(100,2,60), PTS(100,2), S, SUMX2, SUMX,
     + SUMY, SUMXY, SLOPE(60,15), YINT(60,15),
     + SIGMAB(60,15), SIGMAM(60,15), DVDND, SIGMAY(60,15),
     + XMEAN, YMEAN, R, XVAR, YVAR, VARXY

      DATA SIGMAY/900*0.0/
      DATA SIGMAB/900*0.0/
      DATA SIGMAM/900*0.0/
      DATA SLOPE/900*0.0/
      DATA YINT/900*0.0/
      XVAR = 0.0
      YVAR = 0.0
      VARXY = 0.0

      IF ( JMAX .GT. 15 ) JMAX = 15

      DO 1500 MIN = JMIN, JMAX
         DO 1000 MAX = MIN, JMAX

            SUMX = 0.0
            SUMY = 0.0
            SUMX2 = 0.0
            SUMXY = 0.0
            TRUPTS = 0
            DO 30 I=1,DPTS
               PTS(I,1) = 999.0
               PTS(I,2) = 999.0
30          CONTINUE

            DO 200 J=MIN,MAX
               DO 100 I=1, DPTS
                  IF (S * MINPTS(I,2,J) .LT. PTS(I,2)) THEN
                     PTS(I,1) = MINPTS(I,1,J)
                     PTS(I,2) = MINPTS(I,2,J)
                  ENDIF
100            CONTINUE
200         CONTINUE

            DO 300 I=1, DPTS
               IF ( PTS(I,1) .NE. 999.0 ) THEN
                  IF ((MIN .EQ. JMIN) .AND. (MAX .EQ. JMAX))
```

```
     2            THEN
                     WRITE(19,9005) PTS(I,1), ',', PTS(I,2)
                  ENDIF
                  SUMX = SUMX + PTS(I,1)
                  SUMX2 = SUMX2 + PTS(I,1) * PTS(I,1)
                  SUMY = SUMY + PTS(I,2)
                  SUMXY = SUMXY + PTS(I,1) * PTS(I,2)
                  TRUPTS = TRUPTS + 1
               ENDIF
300         CONTINUE

            IF ((MIN .EQ. JMIN) .AND. (MAX .EQ. JMAX)) THEN
               XMEAN = SUMX / REAL( TRUPTS )
               YMEAN = SUMY / REAL( TRUPTS )
               DO 400 I = 1, DPTS
                  IF ( PTS(I,1) .NE. 999.0 ) THEN
                     VARXY = VARXY + ( PTS(I,1) - XMEAN ) *
     2               ( PTS(I,2) - YMEAN )
                     XVAR = XVAR + ( PTS(I,1) - XMEAN ) ** 2
                     YVAR = YVAR + ( PTS(I,2) - YMEAN ) ** 2
                  ENDIF
400            CONTINUE
               R = VARXY / SQRT( XVAR * YVAR )
            ENDIF
            DVDND = REAL(TRUPTS) * SUMX2 - SUMX * SUMX
            IF ( ( DVDND .NE. 0.0 ) .AND. ( TRUPTS .NE. 1 )
     2      ) THEN
               SLOPE(MIN,MAX) = ( REAL(TRUPTS) * SUMXY -
     2         SUMX * SUMY ) / DVDND
               YINT(MIN,MAX) = ( SUMX2 * SUMY -
     2         SUMX * SUMXY ) / DVDND

               IF ( TRUPTS .GT. 2 ) THEN
                  DO 500 I=1, DPTS
                     IF ( PTS(I,1) .NE. 999.0 ) THEN
                        SIGMAY(MIN,MAX) = SIGMAY(MIN,MAX) +
     2                  ( PTS(I,2) - YINT(MIN,MAX) -
     3                  SLOPE(MIN,MAX) * PTS(I,1) )**2
                     ENDIF
500               CONTINUE
                  SIGMAY(MIN,MAX) = SQRT( SIGMAY(MIN,MAX) /
     2            ( REAL(TRUPTS) - 2.0 ) )
                  SIGMAB(MIN,MAX) = SQRT( SIGMAY(MIN,MAX) *
     2            SIGMAY(MIN,MAX) * SUMX2 / DVDND )
                  SIGMAM(MIN,MAX) = SQRT( REAL(TRUPTS) *
     2            SIGMAY(MIN,MAX) * SIGMAY(MIN,MAX) / DVDND)
               ENDIF
            ENDIF

1000     CONTINUE
1500  CONTINUE

      WRITE(6,9003) ' '
      WRITE(6,9003) '       STD DEV          STD DEV  CORR.'
```

```
      WRITE(6,9003) 'SLOPE   SLOPE   Y-INT   Y-INT   COEF.',
     2 '  POINTS'
      WRITE(6,9015) SLOPE(JMIN,JMAX), SIGMAM(JMIN,JMAX),
     2  YINT(JMIN,JMAX), SIGMAB(JMIN,JMAX), R, TRUPTS
      WRITE(18,9003) 'Slope: Minimum Duration vs. Maximum'
     2 ,' Duration'
      WRITE(18,9014) (I,I=JMIN,JMAX)
      DO 3500 L=JMIN, JMAX
         WRITE(18,9013) L, (SLOPE(L,K),K=JMIN,JMAX)
 3500 CONTINUE

      WRITE(18,9003) 'Y-Intercept: Min. Duration vs. Max.',
     2 ' Duration'
      WRITE(18,9014) (I,I=JMIN,JMAX)
      DO 3700 L=JMIN, JMAX
         WRITE(18,9013) L, (YINT(L,K),K=JMIN,JMAX)
 3700 CONTINUE

      WRITE(18,9003) 'Error in Y: Min Duration vs. Max',
     2 ' Duration'
      WRITE(18,9014) (I,I=JMIN,JMAX)
      DO 3900 L=JMIN, JMAX
         WRITE(18,9013) L, (SIGMAY(L,K),K=JMIN,JMAX)
 3900 CONTINUE

      WRITE(18,9003) 'Error in Slope: Duration vs. Max',
     2 ' Duration'
      WRITE(18,9014) (I,I=JMIN,JMAX)
      DO 4100 L=JMIN, JMAX
         WRITE(18,9013) L, (SIGMAM(L,K),K=JMIN,JMAX)
 4100 CONTINUE

      WRITE(18,9003) 'Error in Y-int: Min Duration vs. Max',
     2 ' Duration'
      WRITE(18,9014) (I,I=JMIN,JMAX)
      DO 4400 L=JMIN, JMAX
         WRITE(18,9013) L, (SIGMAB(L,K),K=JMIN,JMAX)
 4400 CONTINUE

 9001 FORMAT(A)
 9003 FORMAT(1X,A,A)
 9005 FORMAT(1X,F8.2,A,F8.2)
 9011 FORMAT(1X,F5.2,2X,F5.2,3(4X,F5.2))
 9012 FORMAT(I2)
 9013 FORMAT(1X,I2,30(2X,F6.2))
 9014 FORMAT(7X,30(I2,6X))
 9015 FORMAT(1X,F5.2,3X,F5.2,3(2X,F6.2),4X,I3)

 9999 RETURN
C     end of usbroutine STATS
      END
```

7.5.1.12 Program MELT

```
C  PROGRAM MELT   ********************************************
C                                                             *
C    Purpose: to find find the melting layers in              *
C         soundings.                                          *
C                                                             *
C    Programmed by Mark Bourassa  Sept 1988                   *
C                                                             *
C*************************************************************

      LOGICAL FOUND, CONDN
      INTEGER UADATE(30,4), JUA, UACNT, I, J, K, ULAYER(20),
     2 LLAYER(20), UACASE, EXTRAP
      REAL UADATA(30,15), DATA(20,6)
      CHARACTER HEADNG*1, CHOICE*1

   50 WRITE(6,9001) 'CHOOSE:  A) Theoretical critical slope'
      WRITE(6,9001) '         B) Sfc experimental critical',
     2    ' slope'
      WRITE(6,9001) '         C) Standard Melting Layer'
      READ(5,9005) CHOICE

      FOUND = .FALSE.
      JUA = 1
      READ(7,9001) HEADNG
      READ(7,9001) HEADNG
      READ(7,9001) HEADNG
      READ(7,9002,END=9999) (UADATE(JUA,I),I=1,4),
     2 (UADATA(JUA,I),I=1,15)
      JUA = 2
      UACNT = 5

  100 IF ( UACNT .EQ. 1 ) THEN
         READ(7,9001) HEADNG
         READ(7,9001) HEADNG
         READ(7,9001) HEADNG
         UACNT = 4
      ENDIF
      READ(7,9002,END=9999) (UADATE(JUA,I),I=1,4),
     2 (UADATA(JUA,I),I=1,15)
      UACNT = UACNT + 1
      IF ( UACNT .EQ. 61) UACNT = 1

      CALL DTCOMP( UADATE(JUA-1,1), UADATE(JUA-1,2),
     2 UADATE(JUA-1,3), UADATE(JUA-1,4), UADATE(JUA,1),
     3 UADATE(JUA,2), UADATE(JUA,3), UADATE(JUA,4),UACASE,0)

      IF ( UACASE .EQ. 0 ) THEN
         JUA = JUA + 1
         GOTO 100
      ELSE
         K = 1
         JUA = JUA + 1
```

```
      DO 400 I=1,15
         UADATA(JUA,I) = UADATA(JUA-1,I)
400   CONTINUE
      DO 500 I=1,6
         UADATA(JUA-1,I) = UADATA(JUA-2,I+6)
500   CONTINUE
      DO 600 I=1,4
         UADATE(JUA,I) = UADATE(JUA-1,I)
         UADATE(JUA-1,I) = UADATE(JUA-2,I)
600   CONTINUE

      DO 1000 J=1, JUA-1
         IF ( (CHOICE .EQ. 'A') .OR. (CHOICE .EQ. 'a') )
     2   THEN
            CONDN = (( UADATA(J,4) .GE. ( 100.0 - 12.65 *
     2      UADATA(J,3) )) .AND.
     2       ( UADATA(J,3) .NE. -9999.9 ) )
         ELSEIF ( (CHOICE .EQ. 'A') .OR.
     2      (CHOICE .EQ. 'a') ) THEN
            CONDN = ( ( UADATA(J,4) .GE. ( 100.0 - 4.65 *

     2      UADATA(J,3) )) .AND.
     2       ( UADATA(J,3) .NE. -9999.9 ) )
         ELSE
            CONDN = ( UADATA(J,3) .GT. 0.0 )
         ENDIF
         IF ( CONDN ) THEN
            IF ( .NOT. FOUND ) THEN
               IF ( J .EQ. 1 ) THEN
                  ULAYER(K) = INT( UADATA(J,2) )
               ELSE
                  ULAYER(K) = EXTRAP( UADATA(J-1,1),
     2            UADATA(J-1,2), UADATA(J-1,3),
     3            UADATA(J-1,4), UADATA(J,1),UADATA(J,2),
     4            UADATA(J,3), UADATA(J,4) )
               ENDIF
               DATA(K,5) = -99999.0
               DATA(K,6) = 0.0
               IF ( ( UADATA(J-1,1) .NE. -99999.0 ) .AND.
     2         ( UADATA(J,1) .NE. -99999.0 ) .AND.
     3         ( UADATA(J-1,3) .NE. -9999.9 ) .AND.
     4         ( UADATA(J,3) .NE. -9999.9 ) .AND.
     5         ( UADATA(J-1,4) .NE. -99999.9 ) .AND.
     6         ( UADATA(J,4) .NE. -99999.9 ) ) THEN
                  DATA(K,3) = UADATA(J,3) +
     2            ( UADATA(J-1,3) - UADATA(J,3) ) *
     3            ( ULAYER(K) - UADATA(J,1) ) /
     3            ( UADATA(J-1,1) - UADATA(J,1) )
                  DATA(K,4) = UADATA(J,4) +
     2            ( UADATA(J-1,4) - UADATA(J,4) ) *
     3            ( ULAYER(K) - UADATA(J,1) ) /
     3            ( UADATA(J-1,1) - UADATA(J,1) )
                  IF ( DATA(K,3) .GT. DATA(K,5) )
     2            DATA(K,5) = DATA(K,3)
```

```
                  DATA(K,6) = DATA(K,6) +
     2            ( UADATA(J,1) - ULAYER(K) )
     2            * ( DATA(K,3) + UADATA(J-1,3) ) / 2.0
               ELSE
                  DATA(K,3) = -99999.0
                  DATA(K,4) = -99999.0
               ENDIF
               FOUND = .TRUE.
            ELSE
               IF ( UADATA(J,3) .GT. DATA(K,5) )
     2         DATA(K,5) = UADATA(J,3)
               DATA(K,6) = DATA(K,6) +
     2          ( UADATA(J,1) - UADATA(J-1,1))
     2         * ( UADATA(J,3) + UADATA(J-1,3) ) / 2.0
            ENDIF
         ELSEIF ( FOUND ) THEN
            IF ( J .EQ. JUA - 1 ) THEN
               LLAYER(K) = INT( UADATA(J,2) )
            ELSE
               LLAYER(K) = EXTRAP( UADATA(J-1,1),
     2         UADATA(J-1,2), UADATA(J-1,3),
     3         UADATA(J-1,4), UADATA(J,1), UADATA(J,2),
     4         UADATA(J,3), UADATA(J,4) )
            ENDIF
            IF ( ( UADATA(J-1,1) .NE. -99999.0 ) .AND.
     2      ( UADATA(J,1) .NE. -99999.0 ) .AND.
     3      ( UADATA(J-1,3) .NE. -9999.9) .AND.
     4      (UADATA(J,3) .NE. -9999.9) .AND.
     5      ( UADATA(J-1,4) .NE. -99999.9 ) .AND.
     5      ( UADATA(J,4) .NE. -99999.9 ) ) THEN
               DATA(K,1) = UADATA(J,3) + (UADATA(J-1,3) -
     2         UADATA(J,3) ) * (LLAYER(K) - UADATA(J,1) /

     3         ( UADATA(J-1,1) - UADATA(J,1) )
               DATA(K,2) = UADATA(J,4) + (UADATA(J-1,4) -
     2         UADATA(J,4)) * (LLAYER(K) - UADATA(J,1)) /
     3         ( UADATA(J-1,1) - UADATA(J,1) )
               IF ( DATA(K,1) .GT. DATA(K,5) )
     2         DATA(K,5) = DATA(K,1)
               DATA(K,6) = DATA(K,6) +
     2         ( ULAYER(K) - UADATA(J-1,1) )
     3         * ( DATA(K,1) + UADATA(J-1,3) ) / 2.0
               IF ((DATA(K,6) .NE. 0.0 ) .AND. (ULAYER(K)
     2            .NE. LLAYER(K) ) ) THEN
                  DATA(K,6) =  DATA(K,6) /
     2            ( LLAYER(K) - ULAYER(K) )
               ELSE
                  DATA(K,6) = -99999.0
               ENDIF
            ELSE
               DATA(K,1) = -99999.0
               DATA(K,2) = -99999.0
            ENDIF
            K = K + 1
```

```
            FOUND = .FALSE.
         ENDIF
1000     CONTINUE

         IF ( FOUND ) THEN
            LLAYER(K) = INT( UADATA(JUA-1,1) )
            FOUND = .FALSE.
            IF ( UADATA(JUA-1,3) .NE. -9999.9 ) DATA(K,1) =
     2      UADATA(JUA-1,3)
            IF ( UADATA(JUA-1,4) .NE. -9999.9 ) DATA(K,2) =
     2      UADATA(JUA-1,4)
            IF ( UADATA(JUA-1,3) .GT. DATA(K,5) )
     2      DATA(K,5) = UADATA(JUA-1,3)
            IF ( ( LLAYER(K) .NE. ULAYER(K) ) .AND.
     2      DATA(K,6) .NE. 0.0 ) THEN
               DATA(K,6) = DATA(K,6)/(LLAYER(K) - ULAYER(K))
            ELSE
               DATA(K,6) = -99999.0
            ENDIF
            K = K + 1
         ENDIF

         IF ( K .EQ. 1 ) THEN
            WRITE(8,9006) (UADATE(1,I),I=1,4), 'Not found'
         ELSEIF ( K .EQ. 2 ) THEN
            WRITE(8,9007) (UADATE(1,I),I=1,4), LLAYER(1),
     2      'to', ULAYER(1), (DATA(1,I),I=1,6)
         ELSE
            WRITE(8,9007) (UADATE(1,I),I=1,4), LLAYER(1),
     2      'to',ULAYER(1), (DATA(1,I),I=1,6)
            DO 2000 J=2, K-1
               WRITE(8,9008) LLAYER(J),'to',ULAYER(J),
     2         (DATA(J,I),I=1,6)
2000        CONTINUE
         ENDIF

         DO 3000 I = 1, 4
            UADATE(1,I) = UADATE(JUA,I)
3000     CONTINUE

         DO 4000 I = 1, 15
            UADATA(1,I) = UADATA(JUA,I)
4000     CONTINUE

         JUA = 2
         GOTO 100
      ENDIF
9001  FORMAT(1X,A,A,A)
9002  FORMAT(1X,3(I2,1X),I2,F6.2,F7.0,F7.1,3F7.0,F7.2,F7.0,
     2 F7.1,4F7.0,2F7.2)
9003  FORMAT(1X,2F10.3)
9004  FORMAT( 1X,A,I2,A,I2,A,I2,A,I2,A )
9005  FORMAT( A )
9006  FORMAT( 4(1X,I2),3X,A)
```

```
9007 FORMAT( 4(1X,I2),3X,I6,1X,A,1X,I6,6(2X,F8.1) )
9008 FORMAT( 15X,I6,1X,A,1X,I6,6(2X,F8.1) )
9999 STOP
     END
```

7.5.1.13 Function EXTRAP

```
      FUNCTION EXTRAP( P1,X1,TA1,RH1,P2,X2,TA2,RH2 )
      INTEGER EXTRAP
      REAL P1, X1, TA1, RH1, P2, X2, TA2, RH2, X

      IF (( P1 .NE. -999.99 ) .AND. ( P2 .NE. -999.99) .AND.
     2 ( TA1 .NE. -9999.9 ) .AND. ( TA2 .NE. -9999.9 ) .AND.
     3 ( RH1 .NE. -99999.0) .AND. (RH2 .NE. -99999.0 )) THEN
          X = P2 + (100.0 - RH2 - 12.65 * TA2) * (P1 - P2 ) /
     2    ( RH1 - RH2 + 12.65 * ( TA1 - TA2 ) )
      ELSE
          X = -99999.0
      ENDIF
      EXTRAP = INT( X )

      RETURN
      END
```

7.5.2 Extreme annual accretion model programs

The program EXTREMES was used to produce the maximum annual values for (wet snow) accretion mass, vertical load, horizontal load due to the hourly average wind speed, and the horizontal load due to gusting winds. It can be used to determine the annual extremes for transmission lines of any diameter, span length, and torsional stiffness. These variables can be set interactively. Another option is the number of years of annual extremes. There is no upper limit to the number of years. When the program was run in an IBM model 50, with a math coprocessor, it would complete sixteen years of extremes in approximately one hour.

7.5.2.1 Program EXTREMES

```
C     PROGRAM EXTREMES.FOR    ********************************
C                                                              *
C     Purpose: to determine the maximum annual forces          *
C              applied transmission lines due to wet snow      *
C              accretions. Line dynamics (movement) is         *
C              not considered in this model. This serves       *
C              as a shell to run a modified version of         *
C              Dr. Finstad's accretion program OMNICYL, or     *
C              any other suitably modified accretion           *
C              program.                                        *
C                                                              *
C     Definitions:                                             *
C              DSEED  - input for the random number            *
C                       generator RAND                         *
C              DUMMY  - a temperary storage variable           *
C              EGLOAD - extreme horizontal load, in a          *
C                       wet snow event, due to gust [N/m]      *
C              ETMASS - extreme accreted mass in a wet         *
C                       snow event [kg/m]                      *
C              EVLOAD - extreme vertical load in a             *
C                       wet snow event: the weight of the      *
C                       snow [N/m]                             *
C              EWLOAD - extreme horizontal load, in a          *
C                       wet snow event, due to the hourly      *
C                       average wind speed [N/m]               *
C              EVENT  - counter for the number of              *
C                       potential wet snow accretion events    *
C                       (EVENTS) in a year.                    *
C              EVENTS - number of potential wet snow           *
C                       accretion events in a year             *
C              GAUSS  - a probibility function returning       *
C                       a randomly determined number of        *
C                       standard deviations from the mean      *
C                       based on a normal distribution         *
C              RAND   - a function returning a random          *
C                       number between zero and one            *
C                       inclusive.                             *
C              RESPON - the user entered response to a         *
C                       yes/no question                        *
C              XGLOAD - annual extreme horizontal load,        *
C                       in wet snow events, due to gusts       *
C                       [N/m]                                  *
C              XTMASS - anual extreme accreted mass in         *
C                       wet snow events [kg/m]                 *
C              XVLOAD - annual extreme vertical load in        *
C                       wet snow events: the weight of the     *
C                       snow [N/m]                             *
C              XWLOAD - annual extreme horizontal load,        *
C                       in wet snow event, due to the          *
C                       hourly average wind speed [N/m]        *
C              YEAR   - counter for the number of years        *
C                       of extremes that are to be             *
```

```
C                     generated                                        *
C               YEARS  - the number of years of extremes             *
C                      that are to be generated                       *
C                                                                     *
C     I/O Streams:                                                    *
C               5 - keyboard. Warning: opening this file              *
C                   will empty the file                               *
C               6  - screen                                           *
C               11 - file extremes.out                                *
C                                                                     *
C     Subroutines: ACCRETE, GAUSS, KRON, OMNICYL, RAND,               *
C               ROTATE, SMOOTH, XTINPT                                *
C                                                                     *
C     Programmed by Mark Bourassa                                     *
C     University of Alberta                                           *
C     Edmonton, Alberta, Canada                                       *
C     May, 1989                                                       *
C                                                                     *
C **********************************************************************

      DOUBLE PRECISION DSEED, DUMMY, RAND, GAUSS

      REAL ETMASS, EVLOAD, EWLOAD, EGLOAD,
     + XTMASS, XVLOAD, XWLOAD, XGLOAD,
     + RANDOM

      INTEGER YEARS, YEAR, EVENTS, EVENT

      CHARACTER RESPON*1

C     open the non-predefined I/O stream.
      OPEN(UNIT=11,FILE='EXTREMES.OUT',STATUS='UNKNOWN')

C     ask the user to enter the number of years for which
C     the program must generate data
  100 WRITE(6,*) 'ENTER THE NUMBER OF YEARS (INTEGER):'
      READ(5,9001) YEARS
C     check to make sure the entry was correct, if not
C     ask for a new number
      WRITE(6,9008) 'You have entered', YEARS, 'is this',
     2 ' acceptable?'
      READ(6,9009) RESPON
      IF ( ( RESPON .EQ. 'N' ) .OR. ( RESPON .EQ. 'n' ) )
     2 GOTO 100

C     ask the user to enter any number to be used as a
C     seed for the random number generator
      WRITE(6,*) 'ENTER THE SEED FOR THE RANDOM NUMBER',
     2 ' GENERATER'
      READ(5,*) DSEED

C     Call the input routine. This routine is a modified
C     version of INPUT, a subroutine of Dr. Finstad's
C     OMNICYL program.
```

```
      CALL XTINPT

      DO 5000 YEAR=1,YEARS
C     Determine the number of events in the year.
          DUMMY = 22.632D0 + GAUSS(DSEED) * 7.946D0
          EVENTS = INT( DUMMY )
          WRITE(6,9007) 'YEAR ', YEAR, ' HAS ', EVENTS,
     2    ' EVENTS'
C     set the annual extremes to be equal to zero
          XTMASS = 0.0
          XVLOAD = 0.0
          XWLOAD = 0.0
          XGLOAD = 0.0
          DO 4000 EVENT=1,EVENTS
             WRITE(6,9007) 'YEAR # ',YEAR,' EVENT # ',EVENT
C     for each event call OMNICYL: the accretion program
             CALL OMNICYL( ETMASS, EVLOAD, EWLOAD, EGLOAD,
     2       DSEED )
C     check the maximum values from the potential wet snow
C     event. If one is greater than the greatest that has
C     been found yet in the year, then the annual extreme
C     is reset to the greater value.
             IF ( EUMASS .GT. XTMASS ) XTMASS = ETMASS
             IF ( EVLOAD .GT. XVLOAD ) XVLOAD = EVLOAD
             IF ( EWLOAD .GT. XWLOAD ) XWLOAD = EWLOAD
             IF ( EGLOAD .GT. XGLOAD ) XGLOAD = EGLOAD
 4000    CONTINUE
C     after each year write the yearly maximums to the
C     EXTREMES.OUT
         WRITE(11,9010) XTMASS, XVLOAD, XWLOAD, XGLOAD
 5000 CONTINUE

 9001 FORMAT(I3)
 9007 FORMAT(1X,A,I3,A,I3,A)
 9008 FORMAT( 1X,A,1X,I3,1X,A,A)
 9009 FORMAT( A )
 9010 FORMAT( 1X,4(F8.3,2X) )
C     end of program EXTREMES
      END
```

7.5.2.2 Subroutine ACCRETE

```
C     ACCRETE.FOR
C ***************************************************************
C  Subroutine Accrete - for each layer number J within the *
C  period, accrete calulates the impingement               *
C  parameters for the current diameter, the local          *
C  slope, density and ice thickness for each surface       *
C  pt, and the total mass, mean density and center of      *
C  gravity for the layer.                                  *
C                                                          *
C  Input parameters:  LAYER,TAU                            *
C                                                          *
```

```
C  Output parameters: TAREA, TMASS                                    *
C                                                                     *
C  Variables defined in Doc.for                                      *
C                                                                     *
C *********************************************************************
      SUBROUTINE ACCRETE(LAYER,TAU,TAREA,TMASS)
C
      DOUBLE PRECISION  ALPHA(900),DROPDI,DUR,GUST,LWC,
     +                  LAYRX(2,900),LAYRY(2,900),PRESSA,
     +                  PRECIP,RH,RHO(900),SLOPE(900),
     +                  TEMPA,THICK(900),VEL,WINDD
CC
      DOUBLE PRECISION  ACCRAD,ALPHM,AREA,BETA,BZERO,CNTRX,
     +                  CNTRY,CYLRAD,DIAG,E,ELAPS,FALL,
     +                  FREEZ,LAREA,LDENS,LEN,LMASS,L1,L2,
     +                  L3,L4,LXCG,LYCG,MASS,MR,RHOMAX,
     +                  RSPEED,STIFF,S1,S2,TAILEN,TAU,TEMPS
     +                  TAREA,TMASS,VNORM,VNZERO,VZERO,
     +                  WLOAD,XCG,YCG,SPCBET, PI, HALFPI
C
      REAL              ROTN(2)
C
      INTEGER           DIFF,RPTS,LAYER,PNUM,SOURCE,
     +                  WFLAG,PERIOD,NPTS,STAG,QUAR1,QUAR3
C
      LOGICAL           SMDROP,GLAZE
C
      CHARACTER*40      NAME
C
      COMMON/ICE/       LAYRX,LAYRY,ROTN,RPTS,ACCRAD,
     +                  XCG,YCG,PRECIP,ELAPS,NPTS
     +     /INP1/       DROPDI,DUR,GUST,LWC,PRESSA,RH,
     +                  SOURCE,TEMPA,VEL,WFLAG,WINDD,PNUM
     +     /INP2/       LEN,STIFF,CYLRAD,PERIOD,NAME
C
      PI = 4.D0 * DATAN(1.D0)
      HALFPI = PI / 2.D0
C
C  Determine coordinate number at the stagnation point, and
C  at the accretion edges
C
      STAG = NPTS / 2
      QUAR1 = NPTS / 4
      QUAR3 = 3 * NPTS / 4
C
C  if source is snow, collection efficiency is sticking
C  efficiency, estimated from a rough empirical relation
C  derived from Japanese data
C
      IF (SOURCE .EQ. 2) THEN
      FALL = 1.D0
      VNORM = DSQRT(FALL * FALL + VEL * VEL) *
     2  DSIN(WINDD)
```

```
X     If the ground temperature is less than freezing, and
X     wet snow is occurring, then an temperature inversion
X     aloft is assumed. The average temperature difference
X     under these conditions is 2.3 degrees. June 20, 1989
      IF ( TEMPA .GT. 0.0 ) THEN
         BZERO = 3.8D-2 * TEMPA/(2.0 * ACCRAD * VNORM)
      ELSE
         BZERO = 3.8D-2 * (TEMPA + 2.3D0) /
     2    (2.D0 * ACCRAD * VNORM)
      ENDIF
      IF (BZERO .GT. 1.D0)BZERO = 1.D0
         ALPHM = HALFPI
         GLAZE = .FALSE.
C
C  Snow density is contant throughout the deposit. Its value
C  is estimated from another empirical relation derived from
C  Japanese data.
C
         IF (VNORM .LE. 17.8D0) THEN
            RHOMAX = 50.D0 * VNORM
            IF (RHOMAX .LT. 200.D0) RHOMAX = 200.0
         ELSE
            RHOMAX = 890.D0
         ENDIF
      ENDIF
C
C  Calculate slope and surface angle from finite differences
C  for each point on current surface. If the wind has
C  rotated this step, use a larger interval for the slope in
C  order to smooth over the discontinuity at the accretion
C  limit of the previous layer.
C
      IF (RPTS .GT. 0) THEN
         DIFF = (INT(ROTN(1) * 143.2D0) / 5) + 15
         IF (DIFF .GT. 100) DIFF = 100
      ELSE
         DIFF = 1
      ENDIF
C
      DO 400 I = QUAR1+1, QUAR3
         IF (LAYRY(1,I+DIFF) .EQ. LAYRY(1,I-DIFF)) THEN
            SLOPE(I) = SLOPE(I-1)
         ELSE
            SLOPE(I) = (LAYRX(1,I+DIFF) - LAYRX(1,I-DIFF)) /
     +      DABS(LAYRY(1,I+DIFF) - LAYRY(1,I-DIFF))
         ENDIF
         IF (DABS(SLOPE(I)) .LT. 1.D-8) SLOPE(I) = 0.0
C
C  Calculate angle between slope and current free stream
C  direction
C
      RSPEED = ROTN(1)
      IF (RSPEED .GE. (2.D0 * PI)) RSPEED = RSPEED
     + - (INT(RSPEED/(2.D0 * PI)) * 2.D0 * PI)
```

```
            IF (LAYRY(1,I+DIFF) .LT. LAYRY(1,I-DIFF)) THEN
               ALPHA(I) = DABS(PI - DATAN(SLOPE(I)) - RSPEED)
            ELSE IF (RSPEED .GT. PI) THEN
               IF (LAYRY(1,I+DIFF) .LT. LAYRY(1,I-DIFF)) THEN
                  ALPHA(I) = DABS(PI - DATAN(SLOPE(I)) - RSPEED)
               ELSE
                  ALPHA(I) = DABS(2.D0 * PI + DATAN(SLOPE(I)) -
     2            RSPEED)
               ENDIF
            ELSE
               ALPHA(I) = DABS(DATAN(SLOPE(I)) - RSPEED)
            ENDIF
C
  400    CONTINUE
C
C     Calculate for each point local density, collision
C     efficiency and ice thickness. The method of calculation
C     for Beta depends on the value of SMDROP.
C
      DO     500 I = QUAR1+1, QUAR3
               RHO(I) = RHOMAX *
     +         (1.D0 - .143D0 * (ALPHA(I)/(ALPHM+TAILEN)) -
     +         .246D0 * ((ALPHA(I)/(ALPHM+TAILEN)) ** 2) -
     +         .309D0 * ((ALPHA(I)/(ALPHM+TAILEN)) ** 3))
C
C     Make sure density does not become too small
C
               IF (RHO(I) .LT. 50.D0) RHO(I) = 50.D0
C
C     Calculate local collision efficiency
C
               BETA =SPCBET(ALPHA(I),SMDROP,GLAZE,SOURCE,BZERO,
     +         ALPHM,TAILEN)
C
C     Calculate local ice thickness perpendicular to surface
C     (in non-dimensional units)
C
               THICK(I) = BETA * VNORM * TAU * LWC /
     +         (RHO(I) * CYLRAD)
C
C     Define new layer surface coordinates.
C
               IF (LAYRY(1,I+DIFF) .LT. LAYRY(1,I-DIFF)) THEN
                  LAYRX(2,I) = LAYRX(1,I) + (THICK(I) *
     +            DCOS(DATAN(SLOPE(I))))
               ELSE
                  LAYRX(2,I) = LAYRX(1,I) - (THICK(I) *
     +            DCOS(DATAN(SLOPE(I))))
               ENDIF
               LAYRY(2,I) = LAYRY(1,I) + (THICK(I) *
     +         DSIN(DATAN(SLOPE(I))))
     500 CONTINUE
C
C     Set non-accreting part of surface to be the same as in
```

```
C  the previous layer
C
      DO 525 I = 1,QUAR1
         LAYRX(2,I) = LAYRX(1,I)
         LAYRY(2,I) = LAYRY(1,I)
         LAYRX(2,QUAR3+I) = LAYRX(1,QUAR3+I)
         LAYRY(2,QUAR3+I) = LAYRY(1,QUAR3+I)
  525 CONTINUE
C
C  Calculate mass and mean density of the layer from
C  the sum of areas between surface points on successive
C  layers. First, initialize layer mass and area, and the
C  coordinates for the layer centre of gravity.
C
      LMASS = 0.D0
      LAREA = 0.D0
      LXCG  = 0.D0
      LYCG  = 0.D0
      DO 560 I = QUAR1+1, QUAR3-1
         IF ((THICK(I) .GT. 0.D0).AND.(THICK(I+1)
     2   .GT. 0.D0)) THEN
            L4 = THICK(I) * CYLRAD
            L2 = THICK(I+1) * CYLRAD
            L1 = DSQRT((LAYRX(1,I+1) - LAYRX(1,I))**2 +
     +      (LAYRY(1,I+1) - LAYRY(1,I))**2) * CYLRAD
            L3 = DSQRT((LAYRX(2,I+1) - LAYRX(2,I))**2 +
     +      (LAYRY(2,I+1) - LAYRY(2,I))**2) * CYLRAD
            DIAG = DSQRT((LAYRX(2,I+1)-LAYRX(1,I))**2 +
     +      (LAYRY(2,I+1) - LAYRY(1,I))**2) * CYLRAD
            S1 = 0.5D0 * (DIAG+L1+L2)
            S2 = 0.5D0 * (DIAG+L3+L4)
            AREA = DSQRT(S1 * (S1-DIAG) * (S1-L1) *
     +      (S1-L2)) + DSQRT(S2 * (S2-DIAG) * (S2-L3) *
     +      (S2-L4))
            MASS = ((RHO(I) + RHO(I+1)) / 2.D0) * AREA
            LMASS   = LMASS + MASS
            LAREA   = LAREA + AREA
C
C  Calculate central point of each quad, and its
C  contribution to the layer centre of gravity
C
            CNTRX = 0.25D0 * (LAYRX(2,I) + LAYRX(1,I)
     +      + LAYRX(2,I+1) + LAYRX(1,I+1))
            CNTRY = 0.25D0 * (LAYRY(2,I) + LAYRY(1,I)
     +      + LAYRY(2,I+1) + LAYRY(1,I+1))
            LXCG  = LXCG + (MASS * CNTRX)
            LYCG  = LYCG + (MASS * CNTRY)
         ENDIF
  560 CONTINUE
C
C  Calculate overall centre of gravity including the new
C  layer
C
      IF ( (TMASS + LMASS) .GT. 0.0 ) THEN
```

```
            XCG = ((TMASS * XCG) + LXCG) / (TMASS + LMASS)
            YCG = ((TMASS * YCG) + LYCG) / (TMASS + LMASS)
            IF (LAYER .EQ. 2) YCG = 0.D0
C
C  Calculate mean density.
C
            LDENS = LMASS/LAREA
         ENDIF
C
C  Add mass and area of current layer to the totals
C
         TAREA = TAREA + LAREA
         TMASS = TMASS + LMASS
C
C  Shift surface points closer to stagnation point
C  (this avoids overcrowding or overseparation of surface
C  points as the shape changes).
C
         IF (LAYER .GT. 2)  THEN
            DO 600 I= 1,STAG-1
            LAYRX(2,I) = LAYRX(2,I) + (0.9 * (I/(STAG-1))
        +   * (LAYRX(2,I+1) - LAYRX(2,I)))
            LAYRY(2,I) = LAYRY(2,I) + (0.9 * (I/(STAG-1))
        +   * (LAYRY(2,I+1) - LAYRY(2,I)))
            LAYRX(2,QUAR3-I) = LAYRX(2,QUAR3-I) + (0.9 *
        +   (I/(STAG-1)) *
        +   (LAYRX(2,QUAR3-I-1) - LAYRX(2,QUAR3-I)))
            LAYRY(2,QUAR3-I) = LAYRY(2,QUAR3-I) + (0.9 *
        +   (I/(STAG-1)) *
        +   (LAYRY(2,QUAR3-I-1) - LAYRY(2,QUAR3-I)))
  600    CONTINUE
         ENDIF
C
C  Call subroutine to rotate array indices if required
C
         IF ( TMASS .NE. 0.0D0 ) CALL ROTATE (LAYER,TMASS)
C
C  Call subroutine to smooth the profile by weighted moving
C  averages
C
         CALL SMOOTH
         CALL SMOOTH
C
C  Update radius of cable plus accretion
C
         ACCRAD = 0.5D0 * CYLRAD * (DSQRT((LAYRX(2,QUAR3) -
        + LAYRX(2,QUAR1+1)) ** 2 +
        + (LAYRY(2,QUAR3) - LAYRY(2,QUAR1+1)) ** 2))
         IF (ACCRAD .LT. CYLRAD) ACCRAD = CYLRAD
C
C  Copy new layer points into first array, ready to be the
C  underlying layer for next loop
C
         DO 606 I=1,NPTS
```

```
            LAYRX(1,I) = LAYRX(2,I)
            LAYRY(1,I) = LAYRY(2,I)
  606 CONTINUE
      ROTN(1) = ROTN(2)
C
      RETURN
      END
```

7.5.2.3 Function SPCBET

```
C ***************************************************************
C *        FUNCTION SPCBET - Estimates Beta(alpha) for a        *
C *           simulated droplet size distribution, for a        *
C *           BZERO either less than or greater than 0.65       *
C *           The polynomial function is scaled to (alpha       *
C *           max + tail length), where the tail length         *
C *           is a function of the mvd.                         *
C *           A special distribution for wet, or                *
C *           precipitation icing, is based on the              *
C *           assumption of straight line trajectories.         *
C *                                                             *
C *  Input parameters : ALPHA,ALPHM,BZERO,SMDROP,TAILEN         *
C *                                                             *
C *  Variables defined in DOC.FOR                               *
C *                                                             *
C *  Except for:    X - scaled angle = Alpha /                  *
C *                        (Alphm + Tailen)                     *
C *                                                             *
C ***************************************************************
C
      DOUBLE PRECISION FUNCTION SPCBET(ALPHA,SMDROP,GLAZE,
     + SOURCE,BZERO,ALPHM,TAILEN)
C
      DOUBLE PRECISION ALPHA,ALPHM,BZERO,TAILEN,X,HALFPI
C
      INTEGER          SOURCE
C
      LOGICAL          SMDROP,GLAZE
C
      HALFPI = (4.D0 * DATAN(1.D0)) / 2.D0
C
      X = ALPHA / (ALPHM+TAILEN)
      IF ((GLAZE) .OR. (SOURCE .GT. 1)) THEN
C
C  For wet icing, or precipitation icing, assume dist'n
C  based on straight line trajectories
C
         SPCBET = BZERO * DCOS(X * HALFPI)
      ELSE
C
C  Local c.e. is zero if we are beyond the accretion limit
C
         IF (ALPHA .GE. (ALPHM+TAILEN)) THEN
```

```
            SPCBET = 0.D0
         ELSE
C
C  Local c.e. dist'n based on two typical drop spectra,
C  depending on the value of SMDROP
C
            IF (SMDROP) THEN
               SPCBET = BZERO * (1.D0 - 0.0147D0 * X -
     +         0.488D0 * (X ** 2) - 3.01D0 * (X ** 3) +
     +         2.52D0 * (X ** 4))
            ELSE
               SPCBET = BZERO * (1.D0 + 0.0287D0 * X -
     +         1.936D0 * (X ** 2) + 2.484D0 * (X ** 3) -
     +         4.112D0 * (X ** 4) + 2.538D0 * (X ** 5))
            ENDIF
         ENDIF
      ENDIF
      IF (SPCBET .LT. 0.D0) SPCBET = 0.D0
      IF (SPCBET .GT. BZERO) SPCBET = BZERO
C
      RETURN
      END
```

7.5.2.4 Subroutine OMNICYL

```
C     OMNICYL.FOR  (Main program - CONVERTED TO SUBROUTINE)
X  Converted to subroutine OMNICYL by Mark Bourassa in May
X  1989. The changes made to the code are marked by 'X'
X  comment designators. The subroutine passes the end values
X  of mass/metre (TMASS), the force on the line due to the
X  mass of the accretion, the force due to the average
X  hourly wind velocity, and wind load due to the gust
X  velocity (GLOAD). Most changes removed the I/O
X  statements that detail individual events. The output
X  would be too massive to be practical. The loss of the
X  I/O also greatly increases the speed of the program!
X  Subroutines that do not deal with wet snow have been
X  completely removed.

      SUBROUTINE OMNICYL(ETMASS,EVLOAD,EWLOAD,EGLOAD,DSEED)
C ***************************************************************
C    OMNICYL  - the main program opens output files,            *
C       initalises tables and variables, reads in all           *
C       input, determines the layer time step and controls      *
C       the accretion subroutines for each layer and            *
C       period. When all layers of ice are accreted, the        *
C       results are written to output files. Variable           *
C       dictionary can be found in ICE.DOC                      *
C                                                               *
C ***************************************************************
      DOUBLE PRECISION   DROPDI, DUR, LAYRX(2,900),
     +                   LAYRY(2,900), GUST, LWC, PRECIP,
     +                   PRESSA, RH, TEMPA, VEL, WINDD,
```

```
     +                    TWINDD, VISIB
C
      DOUBLE PRECISION    ACCRAD,BZERO,CD,CYLRAD,ELAPS,FALL,
     +                    GLOAD,LEN,REMAIN,RHCRIT,STIFF,TAU,
     +                    TAREA,TMASS,VLOAD,WLOAD,XCG,YCG,AL,
     +                    VZ,E,SATVAP,FREEZ,TEMPS,PI,HALFPI,
     +                    MR,RHOMAX,TATRND,RHTRND,GAUSS,KRON,
     +                    RANDOM,RAND
C
      REAL                ROTN(2), EVLOAD, EWLOAD, EGLOAD,
     +                    ETMASS
C
      INTEGER             SOURCE, WFLAG
C
      INTEGER             LAYER,RPTS,PERIOD,PNUM,NPTS
C
      LOGICAL             GLAZE
C
      CHARACTER*40NAME
C
      COMMON   /ICE/      LAYRX,LAYRY,ROTN,RPTS,ACCRAD,
     +                    XCG,YCG,PRECIP,ELAPS,NPTS
     +         /INP1/     DROPDI,DUR,GUST,LWC,PRESSA,RH,
     +                    SOURCE,TEMPA,VEL,WFLAG,WINDD,PNUM
     +         /INP2/     LEN,STIFF,CYLRAD,PERIODS,NAME
C
      PI = 4.D0 * DATAN(1.D0)
      HALFPI = PI / 2.D0
C
C  Initialize total mass, area, centre of gravity coords,
C  and rotation counters.
C
      TMASS = 0.D0
      TAREA = 0.D0
      XCG   = 1.D0
      YCG   = 0.D0
      ROTN(1) = 0.D0
      RPTS = 0
C
C  Bare cylinder will be layer no. 1
C
      LAYER = 1
C
C  Set number of points which define the accretion profile
C
      NPTS = 900
C
C  Accretion radius is initially equal to the bare conductor
C  radius
C
      ACCRAD = CYLRAD
C
C  Begin accretion model for each storm period; begins with
C  the bare conductor profile, and continuing each
```

```
C  subsequent period with the accretion profile stored from
C  end of the last period.
C
X     The following lines were added to OMNICYL by
X     Mark Pourassa on May 28, 1989.

X     Set the duration of each accretion period to be one
X     hour
       DUR = 3.6D3
C
C  Fill array of points defining the initial  surface of the
C  cylinder in non-dimensional coordinates. The complete
C  (360 deg) surface contains NPTS equally spaced points.
C  Moved from INPUT. May 28, 1989
C
       STAG = NPTS / 2

X     Determine the duration (PERIOD) of the event

       RANDOM = RAND(DSEED)
X     if the random number is great enough to make the
X     duration of the event three or more hours then
X     proceed. Otherwise set the duration to be equal to
X     zero hours. Events of one or two hours in duration
X     are ignored.
       IF ( RANDOM .GT. 0.68796D0 ) THEN
          DUMMY = 1.0D1 ** ( ( SQRT( 1.0D0 - RANDOM ** 2 )
     2    - 1.0D0 ) * (-1.452) )
          PERIOD = INT( DUMMY + 0.5 )

X     The following was copied from Dr. Finstad's input
X     subroutine (INPUT.FOR)
          LAYRX(1,STAG) = 0.D0
          LAYRY(1,STAG) = 0.D0
          LAYRX(1,NPTS) = 2.D0
          LAYRY(1,NPTS) = 0.D0

          DO 3100 I = 1,STAG-1
             ANGLE = DBLE(I) / DBLE(NPTS)
             LAYRX(1,I) = 1.D0 + DCOS( 2.D0 * PI * ANGLE)
             LAYRY(1,I) = - DSIN(2.D0 * PI * ANGLE)
 3100     CONTINUE
          DO 3200 I = 1,STAG-1
             LAYRX(1,STAG+I) = LAYRX(1,STAG-I)
             LAYRY(1,STAG+I) = -LAYRY(1,STAG-I)
 3200     CONTINUE

X     determine the initial values for the computer
X     generated meteorological parameters
X     determining wind speed
  200     VEL = 4.948D0 + GAUSS(DSEED) * 2.961D0
          IF ( VEL .LT. 0.0D0 ) GOTO 200
X     determining gust speed
          GUST = 1.275 * VEL
```

```
X     determining wind direciton. The average wind
X     direction is perpendicular the line
            WINDD = 9.0D1 + GAUSS(DSEED) * 1.199D2
  225       IF ( WINDD .GT. 3.6D2 ) THEN
               WINDD = WINDD - 360.0D0
               GOTO 225
            ENDIF
  250       IF ( WINDD .LT. 0.0D0 ) THEN
               WINDD = WINDD + 360.0D0
               GOTO 250
            ENDIF
X     ACCRETE is unable to deal with wind directions
X     greater than 180 degrees. Wind directions greater
X     than 180 degress are reduced by 180 degrees
            TWINDD = WINDD * 2.0D0 * PI / 3.6D2
            IF ( TWINDD .LE. PI ) THEN
               WINDD = TWINDD
            ELSE
               WINDD = TWINDD - PI
            ENDIF
X     determine the air temperature. Note that if this is
X     negative it will be assumed (in ACCRETE) that there
X     is an upper level temperature inversion. This
X     justifies the pressence of wet snow under these
X     conditions.
            TEMPA = 2.202D0 + GAUSS(DSEED) * 2.49D0
X     determining the relative humidity
  299       RH = 8.8856D1 + GAUSS(DSEED) * 9.049D0
            IF ( RH .LT. 0.0D0 ) GOTO 299
            IF ( RH .GT. 100.0D0 ) RH = 100.0D0
X     determining the visibility
  400       VISIB = 1.139D1 + GAUSS(DSEED) * 8.751D0
            IF ( VISIB .LE. 0.1D0 ) GOTO 400
            IF ( VISIB .GT. 25.0D0 ) VISIB = 25.0D0
X     determining the rate of precipitation (mm water
X     equivalent per hour) from the visibility
            DUMMY = 0.055 - LOG10( VISIB ) / 0.607
            PRECIP = 1.0D1 ** DUMMY
X     determining the pressure
            PRESSA = ( 9.2869D1 + GAUSS(DSEED) * 0.724D0 ) *
     2      1.0D3
X     determine whether or not the conditions for wet snow
X     are met. If so set SOURCE to equal 2
            IF ( RH .GE. ( 100.0D0 - 12.25D0 * TEMPA ) ) THEN
               SOURCE = 2
X     when the relative humidity is less that the critical
X     relative humidity, but greater than the surface
X     critical relative humidity, then there is a 45%
X     chance of wet snow occurring
            ELSEIF ( ( RH .GE. ( 90.1D0 - 5.3D0 * TEMPA ) )
     2      .AND. ( TEMPA .LT. 1.5D0 ) ) THEN
               RANDOM = RAND(DSEED)
               IF ( RANDOM .LE. 0.45D0 ) THEN
                  SOURCE = 2
```

```
            ELSE
               SOURCE = 3
            ENDIF
         ELSE
            SOURCE = 3
         ENDIF

C   Convert precip rate to flux at the conductor. Assume
C   terminal speed of snowflakes is 1 m/sec, of raindrops,
C   4.5 m/sec.
C
         IF (SOURCE .EQ. 2)  FALL = 1.D0
         IF (SOURCE .EQ. 3)  FALL = 4.5D0
C
C   Compute effective liquid water content
C
         LWC = PRECIP / 3.6D3 / FALL

      ELSE
         PERIOD = 0
      ENDIF
      WRITE(6,*) 'Duration = ', PERIOD

X     Set the extreams for the event to be equal
X     to zero.
      ETMASS = 0.0
      EVLOAD = 0.0
      EWLOAD = 0.0
      EGLOAD = 0.0

X     End of added lines ( 28 May 89)

      DO 20 PNUM=1,PERIOD
C
C  Set initial values of remaining time and elapsed time in
C  the period
C
         REMAIN = DUR
         ELAPS = 0.D0
C
C  Loop for accreting layers begins here
C
  300    CONTINUE
C
C  Calculate layer time step, so layer thickness is
C  roughly 10 percent of current radius.  A density value
C  is assumed based on the icing source, and whether the
C  icing process is wet or dry.
C
X     The next line has been added June 13,1989
         IF ( SOURCE .EQ. 2 ) THEN
            IF ( VEL .GT. 0.0D0 ) THEN
               BZERO = 3.8D-2 * TEMPA / (2.0 * ACCRAD * VEL)
               IF (BZERO .GT. 1.0D0) BZERO = 1.0D0
```

```
                TAU = (20.D0 * ACCRAD / (BZERO * VEL * LWC))
     X  The following line was added 14 June 1989; it sets the
     X  minimum time increment to three minutes (5% of an hour)
                IF ( TAU .LT. 0.05D0 * DUR ) TAU = 0.05D0 *
       2        DUR
              ELSE
                BZERO = 1.0D0
                TAU = DUR
              ENDIF
            ENDIF
     C
     C  For all layers except the last -
     C
            IF ( (TAU .LT. REMAIN) .AND. (SOURCE .EQ. 2) ) THEN
     C
     C  - update remaining time, layer no, and elapsed time
     C
              REMAIN = REMAIN - TAU
              LAYER = LAYER + 1
              ELAPS = ELAPS + TAU
     C
     C  Call subroutine to accrete layer
     C
              CALL ACCRETE(LAYER,TAU,TAREA,TMASS)
     C
     C  Write to screen percentage of time elapsed
     C
              WRITE (6,350) PNUM,INT(ELAPS/DUR * 100.)
      350     FORMAT (' ',' Period ',I2,' is ',I2,' percent',
        2     ' complete')
     C
     C  Continue with next layer in this period
     C
              GO TO 300
     C
     C  Then for the last layer...
     C
            ELSE IF ((TAU .GE. REMAIN) .AND. (SOURCE .EQ. 2))
        2   THEN
              TAU = REMAIN
              LAYER = LAYER + 1
              ELAPS = ELAPS + TAU
              CALL ACCRETE(LAYER,TAU,TAREA,TMASS)
              WRITE (6,360) PNUM
      360     FORMAT (' ',' Period ',I2,' is 100 percent',
        2     ' complete')
            ENDIF
     C
     C  Calculate the vertical mass load due to ice, assuming no
     C  aerodynamic forces, only gravity, and write to MASS file.
     C
            VLOAD = TMASS * 9.81D0
     C
     C  Calculate the mean and max static horizontal wind
```

```
C  loading, assume drag coefficient is 1.1. GUST is the
C  input max'm expected gust speed for the site in question,
C  as C  determined by the user.
C  Assume a drag coefficient = 1.1
C
           CD = 1.1
           WLOAD = (0.5D0 * 1.3D0 * VEL * VEL) * (2.0D0 *
     +     ACCRAD) * CD * DSIN(WINDD) * DSIN(WINDD)
C
           GLOAD = (0.5D0 * 1.3D0 * GUST * GUST) *(2.0D0 *
     +     ACCRAD) * CD * DSIN(WINDD) * DSIN(WINDD)
C
X    check whether or not this hours values are greater than
X    the greatest previous values
           IF ( TMASS .GT. ETMASS ) ETMASS = TMASS
           IF ( VLOAD .GT. EVLOAD ) EVLOAD = VLOAD
           IF ( WLOAD .GT. EWLOAD ) EWLOAD = WLOAD
           IF ( GLOAD .GT. EGLOAD ) EGLOAD = GLOAD

X  if it is not the end of the event then determine and
X  add the changes for the next hour
           IF ( PNUM .LT. PERIOD ) THEN
X Calculate the values for the time related trends
              DUMMY = 0.156D0 - DLOG( DBLE(PNUM) )
              IF ( DUMMY .LT. 0.0D0 ) DUMMY = 0.0D0
              TATRND = -1.0D0 * DSQRT( DUMMY )
              RHTRND = 3.172D0 - 1.955D0 * DLOG( DBLE(PNUM) )
              IF ( RHTRND .LT. 0.0D0 ) RHTRND = 0.0D0

X       Determine and add the hourly changes for each hour.

              TEMPA = TEMPA + KRON(0.34,DSEED) * ( TATRND
     2        + 0.01 + GAUSS(DSEED) * 0.656D0 )
  450         DUMMY = KRON(0.18) * GAUSS(DSEED) * 1.598D0
              VEL = VEL + DUMMY
              IF ( VEL .LT. 0.0D0 ) THEN
                 VEL = VEL - DUMMY
                 GOTO 450
              ENDIF
              GUST = 1.275 * VEL
              TWINDD = TWINDD + KRON(0.39,DSEED) *
     2        GAUSS(DSEED) * 39.469D0 * 2.0D0 * PI / 3.6D2
  470         IF ( TWINDD .GT. 2.0D0 * PI ) THEN
                 TWINDD = TWINDD - 2.0D0 * PI
                 GOTO 470
              ENDIF
  485         IF ( TWINDD .LT. 0.0D0 ) THEN
                 TWINDD = TWINDD + 2.0D0 * PI
                 GOTO 485
              ENDIF
              IF ( TWINDD .LE. PI ) THEN
                 WINDD = TWINDD
              ELSE
                 WINDD = TWINDD - PI
```

```
            ENDIF
500         DUMMY = KRON(0.24,DSEED) * ( RHTRND +
     2      25.384D0 - 0.275D0 * RH + GAUSS(DSEED) *
     2      4.774D0)
            RH = RH + DUMMY
            IF ( RH .LT. 0.0D0 ) THEN
               RH = RH - DUMMY
               GOTO 500
            ENDIF
            IF ( RH .GT. 100.0D0 ) RH = 100.0D0
600         DUMM = KRON(0.25,DSEED) * GAUSS(DSEED) * 6.391D0
            VISIB = VISIB + DUMMY
            IF ( VISIB .LE. 0.1D0 ) THEN
               VISIB = VISIB - DUMMY
               GOTO 600
            ENDIF
            IF ( VISIB .GT. 25.0D0 ) VISIB = 25.0D0
            DUMMY = 0.055 - LOG10( VISIB ) / 0.607
            PRECIP = 1.0D1 ** DUMMY
            PRESSA = PRESSA + KRON(0.17,DSEED) *
     2      GAUSS(DSEED) * 6.5D1
            IF ( RH .GE. ( 100.0D0 - 12.25D0 * TEMPA ) )
     2      THEN
               SOURCE = 2
            ELSEIF ( ( RH .GE. ( 90.1D0 - 5.3D0 * TEMPA ) )
     2      .AND. ( TEMPA .LT. 1.5D0 ) ) THEN
               RANDOM = RAND(DSEED)
               IF ( RANDOM .LE. 0.45D0 ) THEN
                  SOURCE = 2
               ELSE
               SOURCE = 3
               WFLAG = 0
               ENDIF
            ELSE
               SOURCE = 3
               WFLAG = 0
            ENDIF
C
C   Convert precip rate to flux at the conductor. Assume
C   terminal speed of snowflakes is 1 m/sec, of raindrops,
C   4.5 m/sec.
C
            IF (SOURCE .EQ. 2)  FALL = 1.D0
            IF (SOURCE .EQ. 3)  FALL = 4.5D0
C
C   Compute effective liquid water content
C
            LWC = PRECIP / 3.6D3 / FALL

         ENDIF
C
C  End of period loop, continue with next storm period
C
   20 CONTINUE
```

```
  999 RETURN
      END
```

7.5.2.5 Subroutine ROTATE

```
C     ROTATE.FOR  (subroutines Rotate, Smooth)
C ***************************************************************
C  SUBROUTINE ROTATE - shifts array indices of current          *
C        layer by an amount corresponding to the ice           *
C        weight. Additional torque due to the wind is          *
C        ignored                                               *
C                                                              *
C  Input parameters: LAYER,TMASS                               *
C                                                              *
C  Output     : in Common arrays LAYRX,LAYRY,ROTN              *
C                                                              *
C  Variables defined in DOC.FOR                                *
C                                                              *
C  Except for:  K,M    - index limits for shifting             *
C               RTEMP  - array of iterated rotation amounts    *
C               GRAV   - acceleration due to gravity           *
C               CONST  - combined constants                    *
C               ZETA   - initial position angle of the         *
C                        current                               *
C                        centre of gravity of the accretion    *
C               ANGLE  - angle between the current centre      *
C                        of gravity and the current free       *
C                        stream direction                      *
C                                                              *
C ***************************************************************
C
      SUBROUTINE ROTATE (LAYER,TMASS)
C
      DOUBLE PRECISION    LAYRX(2,900),LAYRY(2,900),TX(450),
     +                    TY(450),RTEMP(20), PRECIP
C
      DOUBLE PRECISION    LEN,STIFF,TMASS,XCG,YCG,CYLRAD,
     +                    ANGLE,ZETA,ACCRAD,CONST,DELROT,
     +                    GRAV,ELAPS,PI
C
       REAL               ROTN(2)
C
       INTEGER            LAYER,RPTS,NPTS,PERIOD
C
       CHARACTER*40       NAME
C
       COMMON  /ICE/      LAYRX,LAYRY,ROTN,RPTS,ACCRAD,
     +                    XCG,YCG,PRECIP,ELAPS,NPTS
     +         /INP2/     LEN,STIFF,CYLRAD,PERIOD,NAME
C
C  Define constants
C
```

```
      NPTS = 900
      GRAV = 9.8D0
      ZETA = DABS(DATAN(YCG / (XCG - 1.0D0)))
      PI = 4.D0 * DATAN(1.D0)
C
C  Determine angle between current c of g and the wind
C  direction
C
      IF (XCG .LT. 1.D0) THEN
         IF (YCG .GT. -1.0D-2) THEN
            ANGLE = ROTN(1) - ZETA
         ELSE
            ANGLE = ROTN(1) - (2.D0 * PI - ZETA)
         ENDIF
      ELSE
         IF (YCG .GT. -1.0D-2) THEN
            ANGLE = ROTN(1) - (PI - ZETA)
         ELSE
            ANGLE = ROTN(1) - (PI + ZETA)
         ENDIF
      ENDIF
      IF (LAYER .EQ. 2) ANGLE = 0.0D0
C
C  Calculate graviational torque constant
C
      CONST = TMASS * GRAV * LEN * CYLRAD *
     + DSQRT(((XCG - 1.D0) ** 2) + (YCG ** 2))
C
C  Begin iteration to calculate rotation angle, using
C  Newton-Raphson method. First guess is angle of new
C  c of g, or 0.5
C
      IF (LAYER .EQ. 2) THEN
         RTEMP(1) = 0.5D0
      ELSE
         RTEMP(1) = ANGLE
      ENDIF
C
      I = 1
   10 CONTINUE
      RTEMP(I+1) = RTEMP(I) + ((CONST * DCOS(RTEMP(I)) -
     + STIFF * RTEMP(I)) /
     + (CONST * DSIN(RTEMP(I)) + STIFF))
      IF (DABS(RTEMP(I+1) - RTEMP(I)) .LT. 1.0D-2) THEN
         DELROT = RTEMP(I+1) - ANGLE
      ELSE IF (I .EQ. 19) THEN
         DELROT = RTEMP(I+1) - ANGLE
      ELSE
         I = I+1
         GOTO 10
      ENDIF
C
C  Calculate the appropriate no. of surface points to
C  rotate, and total rotation amount
```

```
C
      IF (DELROT .LT. 0.D0) DELROT = 0.D0
  501 IF (DELROT .GT.  2.0D0 * PI ) THEN
         DELROT = DELROT - 2.0D0 * PI
         GOTO 501
      ENDIF
      IF ((DELROT .GT. 6.D0) .AND. (ROTN(1) .LT. PI))
     2 DELROT = 0.D0
      RPTS = DINT((DABS(DELROT) / (2.D0 * PI)) *
     2 DBLE(NPTS))
      IF (RPTS .EQ. 0) DELROT = 0.D0
      ROTN(2) = ROTN(1) + DELROT
C
C  Save the first increment no. of points from previous
C  layer in temporoary arrays
C
      DO 575 I = 1,RPTS
         TX(I) = LAYRX(2,I)
         TY(I) = LAYRY(2,I)
  575 CONTINUE
C
C  Renumber layer array to face new direction of wind
C
      K = NPTS - RPTS
      M = NPTS - RPTS + 1
      DO 580 I = 1,K
         LAYRX(2,I) = LAYRX(2,I+RPTS)
         LAYRY(2,I) = LAYRY(2,I+RPTS)
  580 CONTINUE
C
C  Add increment number of points from the previous layer
C  to the 'back' end
C
      DO 585 I = M,NPTS
         LAYRX(2,I) = TX(I - M + 1)
         LAYRY(2,I) = TY(I - M + 1)
  585 CONTINUE
C
      RETURN
      END
```

7.5.2.6 Subroutine SMOOTH

```
C*****************************************************************
C SUBROUTINE SMOOTH - smoothes the array of surface points      *
C         using a weighted moving average                       *
C                                                               *
C                                                               *
C Output     : in Common arrays LAYRX,LAYRY                     *
C                                                               *
C Variables defined in DOC.FOR                                  *
C                                                               *
C Except for:          MAX,MAY - moving average sums            *
```

```
C                    PT1,PT2 - first and last point of     *
C                              array to be averaged        *
C                                                          *
C**********************************************************
C
      SUBROUTINE SMOOTH
C
      DOUBLE PRECISION  LAYRX(2,900),LAYRY(2,900),MAX(900),
     +                  MAY(900),PRECIP
C
      DOUBLE PRECISION  ACCRAD,XCG,YCG,ELAPS
C
      REAL              ROTN(2)
C
      INTEGER           RPTS, NPTS, PT1, PT2, STAG
C
      COMMON   /ICE/    LAYRX,LAYRY,ROTN,RPTS,ACCRAD,
     +                  XCG,YCG,PRECIP,ELAPS,NPTS
C
C Form averaged values
C
      NPTS = 900
      STAG = NPTS / 2
      PT1 = (NPTS / 4) - 10
      PT2 = (3 * NPTS / 4) + 10
      DO 12 I = 1,STAG+20
         MAX(I) = (0.277945D0 * LAYRX(2,PT1+I)) +
     +   0.238693D0 * (LAYRX(2,PT1+I-1) + LAYRX(2,PT1+I+1))
     +   + 0.141267D0 * (LAYRX(2,PT1+I-2) +
     +   LAYRX(2,PT1+I+2)) + 0.035723D0 * (LAYRX(2,PT1+I-3)
     +   + LAYRX(2,PT1+I+3)) - 0.026972D0 *
     +   (LAYRX(2,PT1+I-4) + LAYRX(2,PT1+I+4)) - 0.027864D0
     +   * (LAYRX(2,PT1+I-5) + LAYRX(2,PT1+I+5))
     \   MAY(I) = (0.277945D0 * LAYRY(2,PT1+I))
     +   + 0.238693D0 * (LAYRY(2,PT1+I-1)+LAYRY(2,PT1+I+1))
     +   + 0.141267D0 * (LAYRY(2,PT1+I-2)+LAYRY(2,PT1+I+2))
     +   + 0.035723D0 * (LAYRY(2,PT1+I-3)+LAYRY(2,PT1+I+3))
     +   - 0.026972D0 * (LAYRY(2,PT1+I-4)+LAYRY(2,PT1+I+4))
     +   - 0.027864D0 * (LAYRY(2,PT1+I-5)+LAYRY(2,PT1+I+5))
   12 CONTINUE

      DO 20 I = PT1+20,PT2-20
         LAYRX(2,I) = MAX(I-PT1)
         LAYRY(2,I) = MAY(I-PT1)
   20 CONTINUE
C
      RETURN
      END
```

7.5.2.7 Subroutine XTINPT

```
      SUBROUTINE XTINPT
C*****************************************************************
C                                                                *
C    Purpose:  to interactively aquire a physical                *
C              description of transmission line:                 *
C              line diameter, torsional stiffness, and           *
C              the length of the span.                           *
C                                                                *
C    Adapted for subroutine INPUT of Dr. Finstad's               *
C    accretion model.                                            *
C                                                                *
C*****************************************************************

      DOUBLE PRECISION LAYRX(2,900),LAYRY(2,900),PRECIP(60)

      DOUBLE PRECISION ACCRAD,CYLRAD,ELAPS,LEN,STIFF,XCG,
     2                 YCG

      REAL             ROTN(2)

      INTEGER          RPTS, PERIOD, NPTS

      CHARACTER*40     NAME

      COMMON           /ICE/LAYRX,LAYRY,ROTN,RPTS,ACCRAD,
     +                      XCG,YCG,PRECIP,ELAPS,NPTS
     +                 /INP2/LEN,STIFF,CYLRAD,PERIOD,NAME

      WRITE (6,120)
  120 FORMAT (/////,' ************  TRANSMISSION LINE',
     + ' ICING MODEL   ***************',//,20X,
     + ' Version January 1989 ',//,22X,'K. J. Finstad',//,
     + 20X,' Modified and adapted by Mark Bourassa      ',
     + 'May 1989 ',//,
     + ' Please enter parameters to begin accretion: ',//)

C     ask for the diameter of the transmission line
      WRITE (6,*)
      WRITE (6,*) 'Cylinder diameter in metres? '
      READ  (5,*) CYLRAD
      CYLRAD = CYLRAD / 2.0D0

C     ask for the torsional stiffness of the line
      WRITE (6,*)
      WRITE (6,*) 'torsional stiffness in Nm / rad? '
      READ  (5,*) STIFF

C     ask for the distance between poles or towers
      WRITE (6,*)
      WRITE (6,*) 'span length in metres? '
      READ  (5,*) LEN
```

```
   10  CONTINUE
       RETURN
C     end of subroutine XTINPT
       END
```

7.5.3 Statistical programs and subroutines

These statistical programs and subroutines were used in many of the statistical analyses. For input data they often used modified forms of the output of the program METSTAT. METSTAT could have been used to determine may of these statistics. The advantage of these routine is that they are much faster.

7.5.3.1 Subroutine CURFIT

```
C  SUBROUTNE CURFIT   ************************************
C                                                         *
C    PURPOSE: to calculate a 'best fit' curve to          *
C         unordered data, or a least squares fit to       *
C         unordered data. A least squares fitting         *
C         of raw x and y values is performed by           *
C         setting CURVX=1, and CURVY=1.                   *
C                                                         *
C    Definitions:                                         *
C         ALLPTS    - number of data points (i.e. the     *
C              number of x or y values in PTS             *
C         BDOMAN    - number of points that fall          *
C              outside the domain of either the           *
C              x operator or the y operator, for          *
C              the best fitting curve                     *
C         BSGMAB    - standard deviation of the y         *
C              intercept for the best fitting             *
C              curve                                      *
C         BSGMAM    - standard deviation of the slope     *
C                     for the best fitting curve          *
C         BSLOPE    - slope for the best fitting curve    *
C         BTRUPT    - number of data points within the    *
C              the domain of both operators, for          *
C              the best fitting curve                     *
C         BYINT     - y intercept for the best fitting    *
C              curve                                      *
C         CURVX     - maximum index for the x operator    *
C         CURVY     - maximum index for the y operator    *
C         DOMAIN    - number of points that fall          *
C                     outside the domain of either the    *
C                     x operator or the y operator        *
```

```
C          DVDND     - a dividend used in calculating       *
C               several statistics                          *
C          GPTS      - number of valid points to be         *
C               used in statistics                          *
C          I         - a counter                            *
C          J         - a counter                            *
C          K         - a counter                            *
C          MODE      - character array describing           *
C               operators                                   *
C          PCENT     - the percentage of points that        *
C               fall outside the domain of the              *
C               operators                                   *
C          PTS       - data array from main program.        *
C               x values have a second index of 1,          *
C               y values have a second index of 2           *
C          R         - correlation coefficient              *
C          RBEST     - correlation coefficient for the      *
C               best fitting curve                          *
C          RESPON    - user's response to yes/no            *
C               questions                                   *
C          SIGMAB    - standard deviation of the y          *
C                         intercept                         *
C          SIGMAM    - standard deviation of the slope      *
C          SIGMAY    - standard deviation of y              *
C          SKIP      - logical that is true when a data     *
C               point falls outside the domain of           *
C               one of both of the operators                *
C          SLOPE     - slope of the line                    *
C          SUMX      - total of x values                    *
C          SUMXY     - total of the product of each x       *
C               value and its corresponding y               *
C               value                                       *
C          SUMX2     - sum of the squares of x values       *
C          SUMY      - sum of y values                      *
C          SUMY2     - sum of the squares of y values       *
C          TSTPTS    - data array of valid points.          *
C                         x values have a second index of 1, *
C                         y values have a second index of 2 *
C          TRUPTS    - true number of points being          *
C               examined                                    *
C          VARXY     - covariance                           *
C          X         - absolute value of regression         *
C               coefficient                                 *
C          XBEST     - index of best x operator             *
C          XMEAN     - mean of x values                     *
C          XVAR      - variance in x                        *
C          Y         - absolute value of the best           *
C               regression coefficient                      *
C          YBEST     - index of best y operator             *
C          YINT      - y intercept                          *
C          YMEAN     - mean of y values                     *
C          YVAR      - variance in y                        *
C                                                           *
C     Input parameters: ALLPTS, PTS, CURVX, CURVY           *
```

```
C         See above                                                    *
C                                                                      *
C     Output parameters: BSLOPE, BYINT, RBEST                          *
C         See above                                                    *
C                                                                      *
C     I/O streams:                                                     *
C          5 - keyboard                                                *
C          6 - screen                                                  *
C          22 - output file                                            *
C                                                                      *
C     Programmed by Mark Bourassa                                      *
C     Feb 21, 1989                                                     *
C     University of Alberta                                            *
C     Edmonton, Alberta, Canada                                        *
C                                                                      *
C**********************************************************************

      SUBROUTINE CURFIT(ALLPTS,PTS,CURVX,CURVY,BYINT,
     +                  BSLOPE,RBEST)

      REAL SUMX, SUMX2, SUMY, SUMXY, TRUPTS, DVDND, SLOPE,
     +     SIGMAM, YINT, SIGMAB, PTS(1600,2), SUMY2, SIGMAY,
     +     XVAR, YVAR, VARXY, XMEAN, YMEAN, R, X, Y, BSLOPE,
     +     BSGMAM, BYINT, BSGMAB, RBEST, TSTPTS(1600,2),
     +     PCENT
      INTEGER ALLPTS, DOMAIN, BTRUPT, BDOMAN, GPTS, I, J, K,
     +     XBEST, YBEST, CURVX, CURVY
      LOGICAL SKIP
      CHARACTER*12 MODE(7)
      CHARACTER*1 RESPON

C     initialize arrays, set the best correlation
C     coefficient to equal zero
      RBEST = 0.0
      MODE(1) = 'Unmodified  '
      MODE(2) = 'Squared     '
      MODE(3) = 'Square Root '
      MODE(4) = 'Natural Log '
      MODE(5) = 'Log 10      '
      MODE(6) = 'Anti-nat Log'
      MODE(7) = 'Anti-log 10 '

      DO 5000 I=1,CURVX
         DO 4900 J=1,CURVY
C     for each curve initialize the sums at zero
            DOMAIN = 0
            SUMX = 0.0
            SUMY = 0.0
            SUMX2 = 0.0
            SUMXY = 0.0
            SUMY2 = 0.0
            TRUPTS = 0.0
            DVDND = 0.0
            SLOPE = 0.0
```

```
            SIGMAM = 0.0
            YINT = 0.0
            SIGMAB = 0.0
            SIGMAY = 0.0
            XMEAN = 0.0
            YMEAN = 0.0
            VARXY = 0.0
            XVAR = 0.0
            YVAR = 0.0
            GPTS = 1

C     determine if the x value is within the domain of
C     the operator. If so determine the modified x value
            DO 25 K=1,ALLPTS
               SKIP = .FALSE.
               IF ( I .EQ. 1 ) THEN
                  TSTPTS(GPTS,1) = PTS(K,1)
               ELSEIF ( I .EQ. 2 ) THEN
                  TSTPTS(GPTS,1) = PTS(K,1) * PTS(K,1)
               ELSEIF ( I .EQ. 3 ) THEN
                  IF ( PTS(K,1).GT. 0.0 ) THEN
                     TSTPTS(GPTS,1) = SQRT( PTS(K,1) )
                  ELSE
                     SKIP = .TRUE.
                  ENDIF
               ELSEIF ( I .EQ. 4 ) THEN
                  IF ( PTS(K,1).GT. 0.0 ) THEN
                     TSTPTS(GPTS,1) = LOG( PTS(K,1) )
                  ELSE
                     SKIP = .TRUE.
                  ENDIF
               ELSEIF ( I .EQ. 5 ) THEN
                  IF ( PTS(K,1).GT. 0.0 ) THEN
                     TSTPTS(GPTS,1) = LOG10( PTS(K,1) )
                  ELSE
                     SKIP = .TRUE.
                  ENDIF
               ELSEIF ( I .EQ. 6 ) THEN
                  TSTPTS(GPTS,1) = ALOG( PTS(K,1) )
               ELSEIF ( I .EQ. 7 ) THEN
                  TSTPTS(GPTS,1) = ALOG10( PTS(K,1) )
               ENDIF

C     determine if the y value is within the domain of
C     the operator. If so determine the modified y value
               IF ( J .EQ. 1 ) THEN
                  TSTPTS(GPTS,2) = PTS(K,2)
               ELSEIF ( J .EQ. 2 ) THEN
                  TSTPTS(GPTS,2) = PTS(K,2) * PTS(K,2)
               ELSEIF ( J .EQ. 3 ) THEN
                  IF (PTS(K,2) .GT. 0.0 ) THEN
                     TSTPTS(GPTS,2) = SQRT( PTS(K,2) )
                  ELSE
                     SKIP = .TRUE.
```

```
                  ENDIF
               ELSEIF ( J .EQ. 4 ) THEN
                  IF (PTS(K,2) .GT. 0.0 ) THEN
                     TSTPTS(GPTS,2) = LOG( PTS(K,2) )
                  ELSE
                     SKIP = .TRUE.
                  ENDIF
               ELSEIF ( J .EQ. 5 ) THEN
                  IF (PTS(K,2) .GT. 0.0 ) THEN
                     TSTPTS(GPTS,2) = LOG10( PTS(K,2) )
                  ELSE
                     SKIP = .TRUE.
                  ENDIF
               ELSEIF ( J .EQ. 6 ) THEN
                  TSTPTS(GPTS,2) = ALOG( PTS(K,2) )
               ELSEIF ( J .EQ. 7 ) THEN
                  TSTPTS(GPTS,2) = ALOG10( PTS(K,2) )
               ENDIF

C     keep track of the number of valid points and the
C     number of points that were outside the domain of
C     one or both functions
               IF ( .NOT. SKIP ) THEN
                  GPTS = GPTS + 1
               ELSE
                  DOMAIN = DOMAIN + 1
               ENDIF
   25       CONTINUE
            GPTS = GPTS - 1

C     if some points were rejected write the details to the
C     screen and ask it the curve should be rejected
            IF ( DOMAIN .GT. 0 ) THEN
               PCENT = 100.0 * REAL( DOMAIN ) /
     2         REAL( GPTS + DOMAIN )
               WRITE(6,9010) DOMAIN,' points (',PCENT,
     2         '%) are outside) the domain of the ',
     3         'functions!'
               WRITE(6,9001) MODE(I), ' vs. ', MODE(J)
               WRITE(6,*) 'Is this to high? (Y/N)'
               READ(5,9009) RESPON
               IF ( RESPON .EQ. 'Y' ) THEN
                  WRITE(6,*) 'This correlation will be ',
     2            'ignored.'
                  GOTO 4800
                  ENDIF
               ENDIF

C     make sums
   50          DO 100 K=1,GPTS
                  SUMX = SUMX + TSTPTS(K,1)
                  SUMY = SUMY + TSTPTS(K,2)
                  SUMX2 = SUMX2 + TSTPTS(K,1) * TSTPTS(K,1)
                  SUMXY = SUMXY + TSTPTS(K,1) * TSTPTS(K,2)
```

```
                  SUMY2 = SUMY2 + TSTPTS(K,2) * TSTPTS(K,2)
                  TRUPTS = TRUPTS + 1
  100          CONTINUE

C     determine the slope, y intercept, and the associated
C     standard deviations
               DVDND = TRUPTS * SUMX2 - SUMX * SUMX
               IF ( (DVDND .NE. 0.0 ) .AND.
     2         ( TRUPTS .NE. 1.0)) THEN
                  SLOPE = (TRUPTS * SUMXY - SUMX * SUMY) /
     2            DVDND
                  YINT = ( SUMX2 * SUMY - SUMX * SUMXY ) /
     2            DVDND
               ENDIF
               DO 1500 K=1, GPTS
                  SIGMAY = SIGMAY + ( TSTPTS(K,2) - YINT -
     2            SLOPE * TSTPTS(K,1) ) ** 2
 1500          CONTINUE
               IF ( ( TRUPTS .GT. 2.0 ) .AND.
     2         ( DVDND .NE. 0.0 ) ) THEN
                  SIGMAY = SQRT( SIGMAY / ( TRUPTS - 2.0 ) )
                  SIGMAB = SQRT(SIGMAY * SIGMAY * SUMX2 /
     2            DVDND)
                  SIGMAM = SQRT(TRUPTS * SIGMAY * SIGMAY /
     2            DVDND)
               ENDIF

C     determine the means of x and y values
               XMEAN = SUMX / TRUPTS
               YMEAN = SUMY / TRUPTS

C     determine variances
               DO 2200 K = 1, GPTS
                  VARXY = VARXY + ( TSTPTS(K,1) - XMEAN ) *
     2            ( TSTPTS(K,2) - YMEAN )
                  XVAR = XVAR + ( TSTPTS(K,1) - XMEAN ) ** 2
                  YVAR = YVAR + ( TSTPTS(K,2) - YMEAN ) ** 2
 2200          CONTINUE

C     determine correlation coefficient
               IF ( XVAR * YVAR .GT. 0.0 ) THEN
                  R = VARXY / SQRT( XVAR * YVAR )
               ELSE
                  R = -9.99
               ENDIF

C     determine whether this is better than the previous
C     best fit
               IF ( R .GE. -1.0 ) THEN
                  X = ABS( R )
               ELSE
                  X = -1.0
               ENDIF
               Y = ABS( RBEST )
```

```
C     if the new curve is better then store its
C     characteristics
                IF ( X .GT. Y ) THEN
                   BSLOPE = SLOPE
                   BSGMAM = SIGMAM
                   BYINT = YINT
                   BSGMAB = SIGMAB
                   BTRUPT = INT(TRUPTS)
                   BDOMAN = INT(DOMAIN)
                   RBEST = R
                   XBEST = I
                   YBEST = J
                ENDIF

 4800         CONTINUE
 4900      CONTINUE
 5000  CONTINUE

C     write the results to the screen and to the
C     output file
       WRITE(6,9001) ' '
       WRITE(6,9001) 'X becomes ',MODE(XBEST),
      2 '& Y becomes ',MODE(YBEST)
       WRITE(6,9022) 'Slope              ', BSLOPE
       WRITE(6,9022) 'Stand. Deviation', BSGMAM
       WRITE(6,9022) 'Y-Intercept        ', BYINT
       WRITE(6,9022) 'Stand. Deviation', BSGMAB
       WRITE(6,9022) 'Correllatn Coef.', RBEST
       WRITE(6,9021) 'Data Points        ', BTRUPT
       WRITE(6,9021) 'Domain Rejection', BDOMAN

       WRITE(22,9001) 'X becomes ',MODE(XBEST),
      2 '& Y becomes ',MODE(YBEST)
       WRITE(22,9022) 'Slope             ', BSLOPE
       WRITE(22,9022) 'Stand. Deviation', BSGMAM
       WRITE(22,9022) 'Y-Intercept       ', BYINT
       WRITE(22,9022) 'Stand. Deviation', BSGMAB
       WRITE(22,9022) 'Correllatn Coef.', RBEST
       WRITE(22,9021) 'Data Points       ', BTRUPT
       WRITE(22,9021) 'Domain Rejection', BDOMAN

 9001 FORMAT(1X,A,A,A,A,A)
 9002 FORMAT(1X,A,T25,10I3)
 9003 FORMAT(A,A,A)
 9009 FORMAT( A1 )
 9010 FORMAT( 1X,I3,A,F4.1,A,A)
 9021 FORMAT(1X,A,3X,60(4X,I3,2X))
 9022 FORMAT(1X,A,60(2X,F7.3))
 9023 FORMAT(1X,I3,1X,A,A,A)

      RETURN
      END
```

7.5.3.2 Program EVA

```
C     PROGRAM EVA.FOR
C*****************************************************************
C                                                                *
C     Purpose: to perform an extreme value analysis.             *
C          Minimum tolerances as a function of                   *
C          designed life time and of risk are also               *
C          tabled.                                               *
C                                                                *
C     Definitions:                                               *
C          EXTMOM - unmodified extreme value                     *
C                   determined by moments method                 *
C          EXTREM - array of extreme values read in              *
C                   from the input file                          *
C          EXTREG - unmodified extreme value                     *
C                   determined by regression method              *
C          I      - a counter                                    *
C          INPUT  - name of the input file                       *
C          LIFE   - array of the designed life times             *
C                   used to calculate the minimum                *
C                   tolerances as a function of risk             *
C          OUTPUT - name of the output file                      *
C          PROB   - probability of c in one                      *
C                   unit of time                                 *
C          PTS    - array of extremes and reduced               *
C                   variates. Extremes have a second             *
C                   index of 2. Reduced variates have            *
C                   a second index of 1                          *
C          R      - correlation coefficient                      *
C          RETRN  - array of return periods of                   *
C                   interest. These are used in a               *
C                   table                                        *
C          RISK   - array of risks. These are used               *
C                   in a table                                   *
C          RP     - array of return periods                      *
C          RV     - reduced variate                              *
C          SDE    - standard deviation of extremes               *
C          SLOPE  - slope of the best fit line of                *
C                   extremes as a function of the               *
C                   reduced variate                              *
C          VALUES - number of extreme values                     *
C          XMEAN  - mean value of extremes                       *
C          YINT   - y intercept of the best fit                  *
C                   line of extremes as a function              *
C                   of the reduced variate                       *
C                                                                *
C     I/O streams:                                               *
C          5 - keyboard                                          *
C          6 - screen                                            *
C          7 - input file                                        *
C          22 - output file                                      *
C                                                                *
```

```
C     Programmed by Mark Bourassa                                      *
C     June 25, 1989                                                    *
C     University of Alberta                                            *
C     Edmonton, Alberta, Canada                                        *
C                                                                      *
C***********************************************************************

      REAL       EXTREM(100), R, YINT, SLOPE, PTS(1600,2),
     +           DUMMY, XMEAN, SDE, MYINT, MSLOPE, PROB,
     +           RV, EXTREG, EXTMOM, LIFE(8), RISK(11)

      INTEGER    VALUES, I, CURVX, CURVY, RETRN(22)

      CHARACTER INPUT*12, OUTPUT*12

      DATA RETRN     /2,3,4,5,6,7,8,9,10,15,20,25,30,
     +                35,40,45,50,60,70,80,90,100/

      DATA LIFE      /2.0,5.0,10.0,15.0,20.0,25.0,50.0,
     +                100.0/

      DATA RISK      /0.75,0.50,0.40,0.30,0.25,0.20,
     +                0.15,0.10,0.05,0.02,0.01/

C     ask for and open a data file
      WRITE(6,*) 'ENTER THE NAME OF A DATA FILE'
      READ(5,9001) INPUT
      OPEN( UNIT=7, FILE=INPUT, STATUS='UNKNOWN' )

C     ask for and open an output file
      WRITE(6,*) 'ENTER THE NAME OF AN OUTPUT FILE'
      READ(5,9001) OUTPUT
      OPEN( UNIT=22, FILE=OUTPUT, STATUS='UNKNOWN' )

C     read the number of extreme values
      READ(7,*) VALUES

      XMEAN = 0.0
C     read each extreme value and determine the mean
      DO 100 I=1, VALUES
         READ(7,*) EXTREM(I)
         EXTREM(I) = EXTREM(I) * 1.275
         XMEAN = XMEAN + EXTREM(I)
  100    CONTINUE
      XMEAN = XMEAN / REAL( VALUES )

C     determine the standard deviation of the extremes
      SDE = 0.0
      DO 150 I = 1, VALUES
         SDE = SDE + (EXTREM(I) - XMEAN) ** 2
  150 CONTINUE
      SDE = SQRT( SDE / REAL( VALUES - 1 ) )
```

```
C     sort the extreme values
      CALL SORT( EXTREM, VALUES )

      DO 200 I = 1, VALUES
         DUMMY = REAL( I ) / REAL( VALUES + 1 )
         DUMMY = -1.0 * LOG( DUMMY )
         PTS(I,1) = -1.0 * LOG( DUMMY )
         PTS(I,2) = EXTREM(I)
         WRITE(22,*) PTS(I,1), PTS(I,2)
  200 CONTINUE

C     perform a least squares analysis on the data
      CURVX = 1
      CURVY = 1
      CALL CURFIT(VALUES,PTS,CURVX,CURVY,YINT,SLOPE,R)

C     write, to the screen and the output file, the
C     constants describing the relationship between
C     the extremes and the reduced variate. These
C     constants were determined through the regression
C     method
      WRITE(6,9002) 'Regression estimates are:  a = ',
     2    SLOPE
      WRITE(6,9002) '                           u = ',
     2  YINT
      WRITE(22,9002) 'Regression estimates are:  a = ',
     2    SLOPE
      WRITE(22,9002) '                           u = ',
     2  YINT

C     determine and write the constants found by the
C     method of moments
      MSLOPE = SDE * SQRT( 6.0 ) / 3.1415927
      MYINT = XMEAN - MSLOPE * 0.57721566

      WRITE(6,9002) 'Moment estimates are:  a = ', MSLOPE
      WRITE(6,9002) '                       u = ', MYINT
      WRITE(22,9002) 'Moment estimates are:  a = ', MSLOPE
      WRITE(22,9002) '                       u = ', MYINT

C     make a table with return period, probability of
C     c, reduced variate, and extremes calculated
C     by both regression and moment techniques
      WRITE(22,9001) ' Return              Reduced ',
     +    '   Extreme   Values '
      WRITE(22,9001) ' Period  Probability  Variate ',
     +    '     Reg.      Mom. '
      WRITE(22,9001) ' (years)                      ',
     +    '   (     )   (     )'
      DO 1000 I = 1, 22
         PROB = 1.0 / REAL( RETRN(I) )
         RV = -1.0 * LOG( -1.0 * LOG( 1.0 - PROB ) )
         EXTREG = SLOPE * RV + YINT
         EXTMOM = MSLOPE * RV + MYINT
```

```
            WRITE(22,9004) RETRN(I), PROB, RV, EXTREG,
          +   EXTMOM
 1000     CONTINUE

C         make a table of minimum tolerances as a function
C         of risk and the length of time the structure is
C         to remain useable.
          WRITE(22,9001) '          Designed Lifetime'
          WRITE(22,9005) 'Risk', (INT( LIFE(J) ),J=1,8)
          DO 1200 I = 1, 11
             WRITE(22,9006) RISK(I), ( MYINT - MSLOPE *
         2   LOG( -1.0 * LOG( 1.0 - RISK(I) / LIFE(J) ) ),
         3   J=1, 8)
 1200     CONTINUE


 9001     FORMAT( A,A )
 9002     FORMAT( 1X,A,F8.4,3X )
 9004     FORMAT( 3X,I3,6X,F5.3,5X,F6.3,5X,F7.3,3X,F7.3 )
 9005     FORMAT( 1X,A,3X,I3,7(5X,I3) )
 9006     FORMAT( 1X,F4.2,11(2X,F6.2) )
C         end of program EVA
          END
```

7.5.3.3 Program GAUSS

```
C  FUNCTION GAUSS  ****************************************
C                                                          *
C     Purpose: to randomly determine a number of           *
C          standard deviations from a mean, based          *
C          on the probability distribution of a            *
C          normal distribution. In other words to          *
C          determine the number of 'z' values from         *
C          the mean.                                       *
C                                                          *
C     Definitions:                                         *
C          DELTA  - the fraction of a standard             *
C                   deviation between adjacent             *
C                   probabilities on the lookup table      *
C          DSEED  - a seed for the random number           *
C                   generator                              *
C          GAUSSI - a lookup table of probabilities        *
C                   for a normal distribution              *
C          POINT1 - a pointer for the lookup table         *
C          POINT2 - a pointer for the lookup table         *
C          POINT3 - a pointer for the lookup table         *
C                   equal to the average of POINT1 and     *
C                   POINT2 rounded down                    *
C          PROBAB - the probability corresponding to       *
C                   the number standard deviations.        *
C                   This probability is the cumulative     *
C                   probability (for a one sided           *
C                   normal distribution) of a random       *
```

```
C                number being within the to be                          *
C                determined number of standard                          *
C                deviations from the mean.                              *
C          RAND  - a function generating random                         *
C                numbers                                                *
C          RANDOM - a random number                                     *
C          Z     - the number of standard                               *
C                deviations from the mean                               *
C                                                                       *
C***********************************************************************

      DOUBLE PRECISION FUNCTION GAUSS( DSEED )

      DOUBLE PRECISION DSEED, RAND, PROBAB

      REAL DELTA, GAUSSI(305), RANDOM, Z

      INTEGER POINT1, POINT2, POINT3

      DATA GAUSSI/0.0000,0.0080,0.0160,0.0239,0.0319,
     +    0.0399,0.0478,0.0558,0.0638,0.0717,0.0797,0.0876,
     +    0.0955,0.1034,0.1113,0.1192,0.1271,0.1350,0.1428,
     +    0.1507,0.1585,0.1663,0.1741,0.1819,0.1897,0.1974,
     +    0.2051,0.2128,0.2205,0.2282,0.2358,0.2434,0.2510,
     +    0.2586,0.2661,0.2737,0.2812,0.2886,0.2961,0.3035,
     +    0.3108,0.3182,0.3255,0.3328,0.3401,0.3473,0.3545,
     +    0.3616,0.3688,0.3759,0.3829,0.3899,0.3969,0.4039,
     +    0.4108,0.4177,0.4245,0.4313,0.4381,0.4448,0.4515,
     +    0.4581,0.4647,0.4713,0.4778,0.4843,0.4907,0.4971,
     +    0.5035,0.5098,0.5161,0.5223,0.5285,0.5346,0.5407,
     +    0.5467,0.5527,0.5587,0.5646,0.5705,0.5763,0.5821,
     +    0.5878,0.5935,0.5991,0.6047,0.6102,0.6157,0.6211,
     +    0.6256,0.6319,0.6372,0.6424,0.6476,0.6528,0.6579,
     +    0.6629,0.6680,0.6729,0.6778,0.6827,0.6875,0.6923,
     +    0.6970,0.7017,0.7063,0.7109,0.7154,0.7199,0.7243,
     +    0.7287,0.7330,0.7373,0.7415,0.7457,0.7499,0.7540,
     +    0.7583,0.7620,0.7660,0.7699,0.7737,0.7775,0.7813,
     +    0.7850,0.7887,0.7923,0.7959,0.7995,0.8029,0.8064,
     +    0.8098,0.8132,0.8165,0.8198,0.8230,0.8265,0.8293,
     +    0.8324,0.8355,0.8385,0.8415,0.8444,0.8473,0.8501,
     +    0.8529,0.8557,0.8584,0.8611,0.8638,0.8664,0.8690,
     +    0.8715,0.8740,0.8764,0.8789,0.8812,0.8836,0.8859,
     +    0.8882,0.8904,0.8926,0.8948,0.8969,0.8990,0.9011,
     +    0.9031,0.9051,0.9070,0.9090,0.9109,0.9127,0.9146,
     +    0.9164,0.9181,0.9199,0.9216,0.9233,0.9249,0.9265,
     +    0.9281,0.9297,0.9312,0.9328,0.9342,0.9357,0.9371,
     +    0.9385,0.9399,0.9412,0.9426,0.9439,0.9451,0.9464,
     +    0.9476,0.9488,0.9500,0.9512,0.9512,0.9523,0.9534,
     +    0.9545,0.9556,0.9576,0.9586,0.9596,0.9606,0.9615,
     +    0.9625,0.9634,0.9643,0.9651,0.9660,0.9668,0.9676,
     +    0.9684,0.9692,0.9700,0.9707,0.9715,0.9722,0.9729,
     +    0.9736,0.9743,0.9749,0.9756,0.9762,0.9768,0.9774,
     +    0.9780,0.9786,0.9791,0.9797,0.9802,0.9807,0.9812,
     +    0.9817,0.9822,0.9827,0.9832,0.9836,0.9840,0.9845,
```

```
     +    0.9849,0.9853,0.9857,0.9861,0.9865,0.9869,0.9872,
     +    0.9876,0.9879,0.9883,0.9886,0.9889,0.9892,0.9895,
     +    0.9898,0.9901,0.9904,0.9907,0.9909,0.9912,0.9915,
     +    0.9917,0.9920,0.9922,0.9924,0.9926,0.9929,0.9931,
     +    0.9933,0.9935,0.9937,0.9939,0.9940,0.9942,0.9944,
     +    0.9946,0.9947,0.9949,0.9950,0.9952,0.9953,0.9955,
     +    0.9956,0.9958,0.9959,0.9960,0.9961,0.9963,0.9964,
     +    0.9965,0.9966,0.9967,0.9968,0.9969,0.9970,0.9971,
     +    0.9972,0.9973,0.9995,0.99994,0.999993,0.9999994/

C    Randomly determine the number of standard deviations
C    from the mean. Due to the complexity of the integral
C    and the slow speed of the program a table is used.

     PROBAB = RAND(DSEED)
     POINT1 = 0
     POINT2 = 305
 200 POINT3 = INT( REAL(POINT1 + POINT2) / 2.0 )
C    search by halves until the pointers adjacent to the
C    probability are found
     IF ( PROBAB .LT. GAUSSI(POINT3) ) THEN
        POINT2 = POINT3
C    if the probability is greater than the greatest on
C    the lookup table then the number of standard
C    deviations for the maximum tabled value is used
     ELSEIF ( PROBAB .GT. GAUSSI(POINT3) ) THEN
        POINT1 = POINT3
C    if the probability is equal to a value listed on
C    the lookup table then both pointers are set at this
C    value
     ELSE
        POINT1 = POINT3
        POINT2 = POINT3
     ENDIF
C    unless the pointers indicate adjacent values on the
C    table keep searching
     IF ( POINT1 .LT. ( POINT2 - 1 ) ) THEN
        GOTO 200
     ELSE
        IF ( POINT2 .LE. 301 ) THEN
           DELTA = 0.01
        ELSE
           DELTA = 0.5
        ENDIF
C    extrapolate (linearly) between values on the table
        IF ( POINT1 .NE. POINT2 ) THEN
           Z = REAL( POINT3 - 1 ) / 1.0D2 + DELTA *
     2        ( GAUSSI(POINT2) - PROBAB ) /
     3        ( GAUSSI(POINT2) - GAUSSI(POINT1) )
C    if DELTA is 0.5 make the corrections
           IF ( POINT2 .GT. 302 ) Z = Z +
     2         REAL( POINT2 - 302 ) * 0.49
        ELSE
           Z = REAL( POINT3 - 1 ) / 1.0D2
```

```
C     if DELTA is 0.5 make the corrections
            IF ( POINT2 .GT. 301 ) Z = Z +
     2         REAL( POINT2 - 301 ) * 0.49
         ENDIF
      ENDIF

C     randomly determine if the number of standard
C     deviations is above or below the mean
      RANDOM = RAND(DSEED)
      IF ( RANDOM .GT. 0.5 ) Z = -Z
      GAUSS = DBLE( Z )

      RETURN
C     end of subroutine GAUSS
      END
```

7.5.3.4 Program KRON

```
C  FUNCTION  KRON  ********************************************
C                                                              *
C    Purpose: to perform a delta function:                     *
C             if a random number is less than or equal         *
C             ALPHA then KRON is set equal to zero;            *
C             otherwise it is set equal to one.                *
C                                                              *
C    Definitions:                                              *
C         ALPHA  - the fractional chance of KRON               *
C                  being zero.                                 *
C         DSEED  - the input for the random number             *
C                  generator.                                  *
C         RANDOM - the number generated by RAND                *
C         RAND   - function for the generation of              *
C                  uniform random numbers                      *
C                                                              *
C    Programmed by Mark Bourassa                               *
C    University of Alberta                                     *
C    Edmonton, Alberta, Canada                                 *
C    5 April 1989                                              *
C                                                              *
C**************************************************************

      DOUBLE PRECISION FUNCTION KRON( ALPHA, DSEED )

      DOUBLE PRECISION DSEED, RANDOM, RAND

      REAL ALPHA

      RANDOM = RAND(DSEED)
      IF ( RANDOM .LE. ALPHA ) THEN
         KRON = 0.0D0
      ELSE
         KRON = 1.0D0
      ENDIF
```

```
      RETURN
      END
```

7.5.3.5 Program SORT

```
C  SUBROUTINE  SORT  ****************************************
C                                                            *
C    Purpose: to perform a bubble sort.                      *
C                                                            *
C    Definitions:                                            *
C         ENDLST  - the maximum number of items in           *
C              the list that need to be sorted               *
C         I       - a counter                                *
C         DATA    - an array of the items being              *
C              sorted. The number of items is 100            *
C         SORTED  - a logical that is false when             *
C              the data is unsorted                          *
C         TEMP    - a temporary storage location             *
C              used when items on the list are               *
C              being exchanged                               *
C         VALUES  - the number of items in the list          *
C                                                            *
C    Programmed by Mark Bourassa   June 5, 1989              *
C    Adapted from a handout from a Computer Science          *
C    351 course.                                             *
C                                                            *
C************************************************************

      SUBROUTINE SORT( DATA, VALUES )

      REAL DATA(100), TEMP

      INTEGER VALUES, ENDLST, I

      LOGICAL   SORTED

C     set SORTED to be false, and the end of the list to
C     be 100.
      SORTED = .FALSE.
      ENDLST = VALUES

C     while the list is not sorted continue sorting
  100 IF ( ( .NOT. SORTED ) .AND. ( ENDLST .GT. 1 ) ) THEN
C     assume the list is sorted unless it is found
C     otherwise
         SORTED = .TRUE.
         DO 200 I=1, ENDLST - 1
C     if the order of adjacent data is incorrect then
C     switch them and set SORTED as false
            IF ( DATA(I) .GT. DATA(I+1) ) THEN
            TEMP = DATA(I)
            DATA(I) = DATA(I+1)
```

```
            DATA(I+1) = TEMP
            SORTED = .FALSE.
            ENDIF
  200     CONTINUE
C       reduce the number of items in the list that need
C       to be sorted
          ENDLST = ENDLST - 1
          GOTO 100
        ENDIF

        RETURN
C       end of subroutine SORT
        END
```